UNIVERSIDAD COMPLUTENSE DE MADRID
FACULTAD DE CIENCIAS FÍSICAS

TESIS DOCTORAL

Multi-UAS minimum time search in dynamic and uncertain environments

Búsqueda en tiempo mínimo en entornos dinámicos con incertidumbre

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

Sara Pérez Carabaza

Directores

Eva Besada Portas
José Antonio López Orozco

Madrid

AIRBUS

Multi-UAS Minimum Time Search in Dynamic and Uncertain Environments

Búsqueda en Tiempo Mínimo en Entornos Dinámicos con Incertidumbre

by Sara Pérez Carabaza

Supervised by:

Eva Besada Portas

José Antonio López Orozco

Computer Architecture and System Engineering Department

Faculty of Physics

Universidad Complutense de Madrid

May, 2019

UNIVERSIDAD COMPLUTENSE MADRID

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD DE LA TESIS PRESENTADA PARA LA OBTENCIÓN DEL TÍTULO DE DOCTOR

D./Dña. Sara Pérez Carabaza

con número de DNI/NIE/Pasaporte 72084044C, estudiante en el Programa de Doctorado en Físicas,

de la Facultad de Ciencias Físicas de la Universidad Complutense de Madrid, como autor/a de la tesis presentada para la obtención del título de Doctor y titulada:

MULTI-UAS MINIMUM TIME SEARCH IN DYNAMIC AND UNCERTAIN ENVIRONMENTS

BÚSQUEDA EN TIEMPO MÍNIMO EN ENTORNOS DINÁMICOS CON INCERTIDUMBRE

y dirigida por: Eva Besada Portas y José Antonio López Orozco

DECLARO QUE:

La tesis es una obra original que no infringe los derechos de propiedad intelectual ni los derechos de propiedad industrial u otros, de acuerdo con el ordenamiento jurídico vigente, en particular, la Ley de Propiedad Intelectual (R.D. legislativo 1/1996, de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual, modificado por la Ley 2/2019, de 1 de marzo, regularizando, aclarando y armonizando las disposiciones legales vigentes sobre la materia), en particular, las disposiciones referidas al derecho de cita.

Del mismo modo, asumo frente a la Universidad cualquier responsabilidad que pudiera derivarse de la autoría o falta de originalidad del contenido de la tesis presentada de conformidad con el ordenamiento jurídico vigente.

En Madrid, a 24 de abril de 2019

Fdo.: Sara Pérez Carabaza

Firmado digitalmente por Sara Pérez Carabaza
Nombre de reconocimiento (DN): cn=Sara Pérez Carabaza, o=Universidad Complutense, ou, email=sapere04@ucm.es, c=ES
Fecha: 2019.04.11 14:55:21 +02'00'

Esta DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD debe ser insertada en la primera página de la tesis presentada para la obtención del título de Doctor.

Acknowledgements

I admit that before starting this thesis I thought that the life of Phd students consisted of doing their own research and occasionally asking for assistance to their supervisors. To my surprise, during these last years and I have had the opportunity to meet many people from whom I have learned a lot, both professionally and personally. Therefore, I would like to thank all of them.

I would first like to express my deep gratitude to my thesis supervisors Eva Besada Portas and José Antonio López Orozco for their continuous support and the inestimable help and motivation during these years. I could not have imagined having better supervisors for my doctoral studies. Besides, I am especially grateful to Jesús Manuel de la Cruz for the kindness he always showed me and for giving me the opportunity of doing this thesis.

My sincere thanks to all the people that formed part of SAVIER, a challenging adventure from which I have learned a lot. I would also like to express my gratitude to Airbus for funding this thesis.

Besides, I want to acknowledge Bernhard Rinner for giving me the opportunity to do my research stay at Alpen-Adria-Universität and for his generosity and research support shown during my stay.

Moreover, I would also like to thank the experts and members of my committee who were involved in the validation survey for this thesis.

I also want to thank all the people of the Computer Architecture and System Engineering Department. Thanks to all for being so kind to me and for encouraging me

to finish my thesis. In particular, I want to thank my office mates for all the hours spent together. Furthermore, I would like to thank Gonzalo Pajares for giving me the opportunity of continuing researching.

Thanks to my longtime friends from Santander for making me feel that nothing has changed every time we meet. And thanks to all the great people I came across during these years at Madrid, especially to my flatmates for making me laugh everyday. Thanks Laura for all the talks in your car and fun upside down. Thanks Mar for becoming my family at Madrid since I met you at the beginning of this thesis, thanks for your ability to turn any ordinary task into fun. Lastly, thanks Daniela for the great trips and visits during these years.

Thanks to Juan for always being there when I need it, helping me disconnect and for all the great moments we have shared.

Finally, I want to especially thank my family, above all my parents, for their love and unconditional support. Thanks to my mother for always listening to my worries and giving me great advice, you are the best friend that a daughter can expect. Thanks to my father for your support and being always my example of hard work.

Not everyone I should thank is mentioned, but all the mentioned ones matter.

Thank you very much, everyone!

Contents

Abstract

Unmanned Aerial Vehicles (UAVs) have experienced a huge development during the last decades and are nowadays used in many different applications such as defense, fire extinction or search and rescue missions. During the development of these missions, the operators monitor and control the UAVs from the Ground Control Station (GCS) with a level of control that depends of the autonomy of the Unmanned Aerial System (UAS), formed by the UAVs, the GCS and the communication between both. Although the use of multiple UAVs generally improves the performance of the missions, it also increases the operator workload, which may end up causing operator overload (specially in UAS with low autonomy level). Therefore, the operator overload can be avoided by the development of more autonomous UAS and tools that decrease the workload of operators during the mission planning and execution. Following this idea, Airbus proposed an innovation project in collaboration with several Spanish Universities entitled "Situational Awareness Virtual EnviRonment" (SAVIER). This thesis is framed in that project and focuses on Minimum Time Search (MTS) missions, where a target with unknown position and dynamics needs to be found as soon as possible.

The work done during this thesis is encompassed within the research field of theory of optimal search, which probabilistically models the available information about the search scenario and proposes search plans according to it. The potential of probabilistically modelling the uncertain information of the search scenario was proven years ago during the search for the H-bomb lost in Palomares (Spain), which after several days of unsuccessful search was finally found thanks to the use of a proba-

bility model of the possible locations of the bomb. However, the search for a lost target implies much more than building the initial probabilistic models, since solving the whole problem is necessary to provide the observation plan. This is done by means of probabilistic search algorithms, which propose optimal search routes considering the initial information about the search scenario and whose definition of optimal route depends on the type of search missions they were designed for. In the case of probabilistic algorithms that deal with MTS missions (MTS algorithms), their objective is to minimize the time required to find the target. Therefore, due to the critical importance of time, MTS missions have several applications such as the search for survivors or military search missions that may involve danger to the UAVs.

This thesis aims to propose new MTS algorithms and realistic models of the elements involved in search missions. The state of the art in probabilistic search encompasses numerous probabilistic search algorithms, which, due to the high complexity of the problem, generally assume simplifications during the modelling of the problem and do not ensure finding the optimal solution. This thesis solves the MTS problem from a new approach based on Ant Colony Optimization (ACO), proposing two types of algorithmic solutions. We have selected ACO metaheuristics due to the good performance they have shown in a variety of problems and their ability to include problem specific information through the use of constructive heuristics. The inclusion of problem specific information can be advantageous in a problem that requires a balance between the quality of solutions and the computational time required to obtain them.

On the one hand, we propose two MTS algorithms based on a widely used ACO algorithm for discrete optimization problems (Max-Min Ant System), which optimize high-level trajectories of multiple UAVs, specified as sequences of adjacent cells of the discretized search region. Hence, due to the consideration of simplified UAVs motion model, these algorithms are adequate to rotatory-wing UAVs and have the advantage of requiring low computational times. The simulated experiments show that, thanks to the proposed MTS constructive heuristic, the MTS algorithms are able to find better quality solutions faster than other state of the art methods. On the other

hand, this thesis proposes a MTS algorithm based on the ACO metaheuristic for continuous optimization problems (Ant Colony for Real Domains, ACOR), which minimizes the target detection time of the UAVs search trajectories while avoiding overflying forbidden areas and collisions among the UAVs. Thanks to the consideration of a complex UAV dynamic model, the proposed search routes are defined by smooth curve paths adequate for the maneuverability restrictions of fixed-wing UAVs. We propose two strategies in order to deal with the increased complexity of the problem derived from the complex UAV models and multi-objective evaluation criteria. On one side, encouraged by the good results in the discrete version of the problem, we include a percentage of ants that use information from a constructive MTS heuristic to build their path (UAVs search trajectories). On the other side, we use a receding horizon controller strategy (dividing the optimization of the full trajectories into several less complex optimization problems) and propose a new optimization criterion that aims to avoid the locality problems derived from this approach. From the conducted simulated experiments we can conclude that the inclusion of heuristic ants increases the solutions quality and the speed of converge of the algorithm, reaching better results than the MTS algorithm based on a genetic algorithm also proposed during this thesis. Besides, the proposed new objective criterion decreases the locality of the solutions and improves the algorithm performance when the MTS algorithm is implemented within a receding horizon controller.

Last but not least, another of the objectives of the thesis, initially defined by the SAVIER project, was to integrate and test the thesis contributions with a GCS developed by Airbus to control ATLANTE UAV. Therefore, not only have new MTS algorithms been developed, but also they have been integrated with a GCS in order to prove that the techniques proposed in this thesis can be integrated with the functionality of a real UAS. Besides, in order to provide a tool that can be used during the search missions, we have developed a graphical user interface that allows to define the search scenario (considering the input information received from the GCS), optimize the search routes and send back the optimized solutions to the GCS. Moreover,

the developed MTS auxiliary tool has been also integrated within a opensource GCS developed during the project, which allows to control multiple UAVs.

Resumen

Los vehículos aéreos no tripulados (Unmmaned Aerial Vehicle, UAV) han experimentado un gran desarrollo durante las últimas décadas y son utilizados en aplicaciones tan variadas como defensa, extinción de incendios o en misiones de búsqueda y rescate. Durante el desarrollo de estas misiones, los operadores monitorizan y controlan los UAVs desde la estación de control de tierra (Grond Control Station, GCS) con un nivel de control dependiente de la autonomía del sistema aéreo no tripulado (Unmanned Aerial System, UAS), formado por los UAVs, la GCS y el sistema de comunicación entre ambos. Aunque el uso de múltiples vehículos aéreos no tripulados generalmente mejora el rendimiento de las misiones, también aumenta la carga de trabajo del operador, lo que puede causar una sensación de sobrecarga de éste (especialmente en UAS con bajo nivel de autonomía). Esta situación puede evitarse mediante el desarrollo de UAS con mayor nivel de autonomía y herramientas que disminuyan la carga de trabajo de los operadores durante la planificación y ejecución de las misiones. Siguiendo esta idea, Airbus propuso un proyecto de innovación en colaboración con varias universidades españolas titulado "Situational Awareness Virtual EnviRonment" (SAVIER). Esta tesis, enmarcada dentro de ese proyecto, se centra en las misiones de búsqueda de tiempo mínimo (Minimum Time Search, MTS), donde es necesario encontrar lo antes posible un objetivo con una posición y dinámica desconocidas.

El trabajo realizado durante esta tesis se engloba dentro del campo de investigación de la teoría de la búsqueda óptima, que modela probabilístiscicamente la información disponible sobre el escenario de búsqueda y propone soluciones teniendo en

cuenta los modelos probabilísticos. El potencial de modelar probabilísticamente la información incierta del escenario se demostró hace años durante la búsqueda de la bomba-H perdida en Palomares (España), que tras varios días de búsqueda sin éxito finalmente se encontró gracias al uso de un modelo de probabilístico que consideraba las posibles ubicaciones de la bomba. Sin embargo, la búsqueda de un objetivo perdido implica mucho más que la construcción de los modelos probabilísticos iniciales, ya que para resolver el problema es necesario obtener un plan de búsqueda óptimo. Esto se resuelve mediante algoritmos de búsqueda probabilísticos, que proponen soluciones teniendo en cuenta la información inicial sobre el escenario y el tipo de misiones de búsqueda para las que fueron diseñadas. En el caso de algoritmos probabilísticos que tratan con misiones MTS (algoritmos MTS), su propósito es minimizar el tiempo requerido para encontrar el objetivo perdido. Por lo tanto, debido a la importancia crítica del tiempo, las misiones MTS tienen varias aplicaciones, como la búsqueda de supervivientes o misiones militares de búsqueda que pueden implicar peligro para los UAVs.

Esta tesis tiene como objetivo proponer nuevos algoritmos MTS y modelos realistas de los elementos involucrados en las misiones de búsqueda. El estado del arte de la búsqueda probabilística abarca numerosos algoritmos que, debido a la alta complejidad del problema, generalmente asumen simplificaciones durante su formulación y no garantizan que la solución propuesta sea óptima. Esta tesis resuelve el problema de MTS desde un nuevo enfoque basado en la metaheurística de optimización de colonias de hormigas (Ant Colony Optimization, ACO) y que propone dos tipos de soluciones algorítmicas. Hemos seleccionado la metaheurística ACO debido a su aplicación satisfactoria en una gran variedad de problemas de optimización y a su capacidad para incluir información específica del problema mediante el uso de heurísticas constructivas. La inclusión de una heurística ad-hoc puede ser beneficiosa en un problema como MTS, donde es necesario encontrar un buen balance entre la calidad de las soluciones y el tiempo de cómputo requerido para obtenerlas.

Por un lado, proponemos dos algoritmos MTS basados en un algoritmo de optimización de colonia de hormigas ampliamente utilizado para problemas de optimiza-

ción discretos (Max-Min Ant Colony System), que optimiza las trayectorias de alto nivel de múltiples vehículos aéreos no tripulados, especificadas como secuencias de celdas adyacentes de la red utilizada para discretizar el área de búsqueda. Por lo tanto, debido a la consideración de un modelo dinámico de UAV simple y que requiere una alta maniobrabilidad, estos algoritmos son adecuados para los UAVs de ala rotatoria y tienen la ventaja de requerir tiempos de computación bajos. Los experimentos realizados muestran que, gracias a la heurística MTS propuesta, los algoritmos de MTS pueden encontrar soluciones de mejor calidad en tiempos de computación menores que otros algoritmos del estado del arte.

Por otro lado, esta tesis propone un algoritmo MTS basado en la metaheurística ACO para problemas de optimización continuos (Ant Colony Optimization for Real Domains, ACOR), que minimiza el tiempo de detección del objetivo de las trayectorias de búsqueda de UAVs mientras evita que estos sobrevuelen áreas de vuelo prohibidas, además de las posibles colisiones entre UAVs. Gracias a la consideración de un modelo dinámico de UAV complejo, las rutas de búsqueda propuestas quedan definidas por curvas suaves adecuadas para las restricciones de maniobrabilidad de los UAVs de ala fija. Proponemos dos estrategias para hacer frente a la mayor complejidad del problema derivada de la complejidad de los modelos dinámicos de los UAVs y los criterios de evaluación de múltiples objetivos. Por un lado, alentados por los buenos resultados obtenidos en la versión discreta del problema, incluimos un porcentaje de hormigas que usan información de una heurística constructiva especifica del problema MTS para determinar su camino (trayectorias de búsqueda de los UAVs). Por otro lado, usamos la estrategia del control predictivo por modelo (dividiendo la optimización de las trayectorias de búsqueda completas en varios problemas de optimización menos complejos) y proponemos un nuevo criterio de optimización para evitar los problemas de localidad derivados de este enfoque. De las simulaciones realizadas podemos concluir que la inclusión de hormigas heurísticas aumenta la calidad de las soluciones y la velocidad de convergencia del algoritmo, logrando mejores resultados que el algoritmo MTS basado en algoritmos genéticos también propuesto durante la tesis. Además, el criterio de optimización propuesto re-

duce la localidad de las soluciones y mejora el rendimiento del algoritmo de hormigas cuando el algoritmo MTS se implementa siguiendo el enfoque de control predictivo por modelo.

Por último, otro de los principales objetivos de la tesis, inicialmente definido por el proyecto SAVIER, era integrar y probar las contribuciones de la tesis con una GCS desarrollada por Airbus para controlar el UAV ATLANTE. Por lo tanto, no solo se han desarrollado nuevos algoritmos MTS, sino que también se han integrado con una GCS para probar que las técnicas propuestas en esta tesis pueden incorporarse a la funcionalidad de un sistema aéreo no tripulado real. Con el objetivo de proporcionar una herramienta que se pueda utilizar fácilmente durante las misiones de búsqueda, hemos desarrollado una interfaz gráfica de usuario que permite definir el escenario de búsqueda (considerando la información recibida de la GCS), optimizar las rutas de búsqueda y enviar las soluciones propuestas a la GCS. Además, la herramienta también se ha integrado dentro de una GCS de código libre desarrollada durante el proyecto, que permite comprobar el funcionamiento de los algoritmos desarrollados en escenarios que involucren a múltiples UAVs.

List of Figures

List of Tables

Nomenclature

Acronyms

ACO Ant Colony Optimization

ACOR Ant Colony Optimization for Continuous/Real Domains

ACS Ant Colony System

AS Ant System

BN Bayesian Network

BOA Bayesian Optimization Algorithm

CACO Continuous Ant Colony Optimization

CEO Cross Entropy Optimization

CIAC Continuous Interacting Ant Colony

DDS Data Distribution Service

DTR Discounted Time Reward

ET Expected target detection Time

FAA Federal Aviation Administration

FCS Flight Control System

FN False Negative

FP False Positive

GA Genetic Algorithm

GCS Ground Control Station

GUI Graphical Unit Interface

HGM Global Maximum Heuristic

HLM Local Maximum Heuristic

HS Spiral Heuristic

MMAS Max Min Ant System

MPMS Mission Planning and Monitoring System

MTS Minimum Time Search

NATO North Atlantic Treaty Organization

NFZ Non Flying Zone

PDF Probability Density Function

POMDP Partially Observable Markov Decision Process

PS Probabilistic Search

PSO Particle Swarm Optimization

R&T Research & Technology

RBF Recursive Bayesian Filter

RHC Recursive Horizon Controller

SaR Search and Rescue

SAVIER Situational Awareness VIrtual EnviRoment

TSP Traveling Salesman Problem

UAS Unmanned Aerial System

UAV Unmanned Aerial Vehicle

UGV Unmanned Ground Vehicle

US Unmanned System

WISaR WIlderness Search and Rescue

Symbols

A Transition matrix

α ACO pheromone influence parameter

β ACO heuristic influence parameter

$\boldsymbol{C}_u$ Control action domain of UAV u

$\boldsymbol{\tau}$ ACO Pheromone table

ΔT Time interval between consecutive time steps

η ACO heuristic

$\mathscr{A}$ ACOR archive of solutions

ρ ACO pheromone evaporation parameter

$\tilde{b}(v^t)$ Unnormalized belief at time step t

$b(v^t)$ Belief or probability map at time step t

c_u^t Control action of UAV u at time step t position

G_Ω Grid of the rectangular search area with dimensions $w_x \cdot w_y$

L Planner horizon of each optimization step of the receding horizon controller

M Number of artificial ants

m Index of an ant tour solution

N Planning horizon

$P(v^0) = b(v^0)$ Initial probability map or belief

Q Number of optimization steps considered in the receding horizon controller

R Algorithm population size

r Index of a individual of the population of solutions of an algorithm

s_u^t Position of UAV u at time step t

T Total time of the search

t Time index

U Number of UAVs

u UAV index

$z_u^t = \{D, \overline{D}\}$ Target detection and no detection measurements

Chapter 1

Introduction

"Time isn´t the main thing. It's the only thing"

Miles Davis

This chapter starts describing the motivation of the thesis, that aims to provide useful tools/algorithms for target search missions performed by Unmanned Aerial Vehicles (UAVs) with the purpose of reducing operators workload. Next, it describes the main objectives of the thesis and how they are approached. Finally, the main contributions are listed and the thesis structure outlined.

Before proceeding, it is worth noting that this thesis has been funded by Airbus and it belongs to the "Situational Awareness VIrtual EnviRonment" (SAVIER) collaboration project signed by Airbus with several Spanish Universities, whose main purpose is to develop new tools and technologies for their future Ground Control Stations (GCS). Hence, the developments of this thesis will be finally tested within an industrial environment provided by Airbus.

1.1. Motivation

An Unmanned Aerial Vehicle (UAV) is an aircraft without an on-board human operator (Newcome, 2004). They are also referred to as Remotely Piloted Aircraft

System (RPAS), emphasizing the remote control of UAVs, or drones, a name originally derived from the similarity to the male bee buzzing sound that the first UAVs made and which is barely noticeable in modern ones.

UAVs roots date back to World War I when the United States of America and France worked on developing unmanned airplanes and technology, later improved throughout World War II (Newcome, 2004). However, its commercial and civil use did not start until 2005, when military UAVs equipped with infrared sensors were used after Hurricane Katrina. This led to the issue of certificates by the Federal Aviation Administration (FAA) to let military drones be used over civilian skies one year later (Rao et al., 2016). Since then, the use and variety of applications of UAVs have experienced a huge development. UAVs entail a great advantage in dull or dangerous missions and are employed in a variety of applications such as mapping (build up a map of an unknown environment), Search and Rescue (SaR) missions, package delivery or fire extinction (Liu et al., 2014).

UAVs are a component of an Unmanned Aerial Systems (UAS), which include the UAV itself, a ground-based command and supervising center, and the communication system between both. UAV operators monitor and control the mission from the Ground Control Station (GCS), with a degree of control of the mission execution that depends on the autonomy of the UAS. According to (Gupta et al., 2013) we can categorize UAS from lowest to highest autonomy as: remotely control UAS (where the UAVs require constant control of the operator), semi-autonomous UAS (where the UAVs require human control only in some parts of the mission, such as landing and taking-off), and fully autonomous UAS (where the UAVs, in theory, do not require human inputs and the operators only have to monitor the mission).

Multiple UAVs can be controlled from the same GCS, generally allowing a more efficient performance of the mission objectives at the expenses of increasing operators workload, which grows exponentially with the number of UAVs (Perez-Rodriguez et al., 2013). Moreover, excessive workload may stress the operators and decrease the mission performance. This is one of the main reasons that has brought research attention to the development of more autonomous UAS and tools that decrease the

workload of operators during the mission planning and execution. In fact, decreasing operator workload is the main objective of the Airbus research project "Situational Awareness VIrtual EnviRonment" (SAVIER) where this thesis is encompassed.

More concretely, this thesis focuses on Minimum Time Search (MTS) missions, where a target with unknown position and dynamics needs to be found as soon as possible. Depending on the level of uncertainty and importance of time, search missions in uncertain environments can be categorized as: coverage problems (where usually there is little information about the target location and although an efficient coverage of the search region is preferred the time is not critical), Probabilistic Search (PS) missions with limited resources (which typically try to maximize the probability of detecting the target within a limited time), and MTS missions (where is it not only important to find the target with limited resources but also to find it as soon as possible).

Minimum time search has both military and civil applications. On one hand, finding military targets quickly can reduce the chances of being attacked or detected. On the other hand, time is a critical factor in search missions that look for humans beings. In Search and Rescue (SaR) missions that take place after natural disasters like an earthquake, a tsunami or an avalanche time is a critical factor for finding survivors, as well as after an accident such as a shipwreck or airplane crash. For instance, survival rates of earthquake victims continually drop with the delay of help, with a sharp drop off at around 48 hours (Kuhlman et al., 2017). Or, in the case of buried victims by an avalanche the survival rates drop dramatically after 15 minutes (Hoffmann et al., 2006).

MTS algorithms can obtain optimized UAV search routes taking into account the probabilistically modelled information about the search scenario. The potential of using probabilistic models about the uncertain information of the search scenario was proven for the first time years ago during the search for the H-bomb lost in Palomares (Spain). After several days of search without success, the bomb was finally found thanks to the use of a probability map of the possible bomb locations, which was constructed considering the probabilities of the possible causes of the accident and

the information given by a fisherman that witnessed the catastrophe. However, the search for a lost target implies much more than building the initial probability map, as in order to solve the whole problem, it is also necessary to provide the optimal search route. In this thesis we focus on the optimization of the search route, a problem that is not trivial and gets more complex when the searcher has dynamic restrictions, the target is not static or the search is carried out by multiple searchers (UAVs in our case).

Nowadays, search operations are mainly performed by operators, who are in charge of planning the search routes using search patterns such as lawnmower or spiral trajectories. For instance, the public search patterns defined in (Interagency Committee on Search and Rescue, 1991) are used by the U.S Department of Defense in SaR missions and by the U.S. Coast Guard in water-based searches. For some specific search scenarios (e.g. when the initial target location can be modelled with an uniform or gaussian probabibility), adequate search patterns (e.g. lawnmower or spiral trajectories) can be an optimal solution. However, in more complex scenarios it has been proven that search patterns are non optimal routes and that the use of the search trajectories proposed by PS optimization algorithms ensure higher chances of finding the target, which implies higher probabilities of finding survivors in SaR scenarios (Lin and Goodrich, 2009).

1.2. Objectives

The main elements involved during the planning of a search mission are outlined in Figure 1.1. The image on the left shows a snapshot of the mission planner of a GCS, used by the operators to monitor and control the mission, and which contains the mission information defined by a set of waypoints (whose flight profile is shown at the bottom). Besides, the search area (defined with a polygon) is displayed in the image on the right and the UAV and its electro-optic sensor are represented over it. The best route that the UAVs should follow inside the search area in order to detect

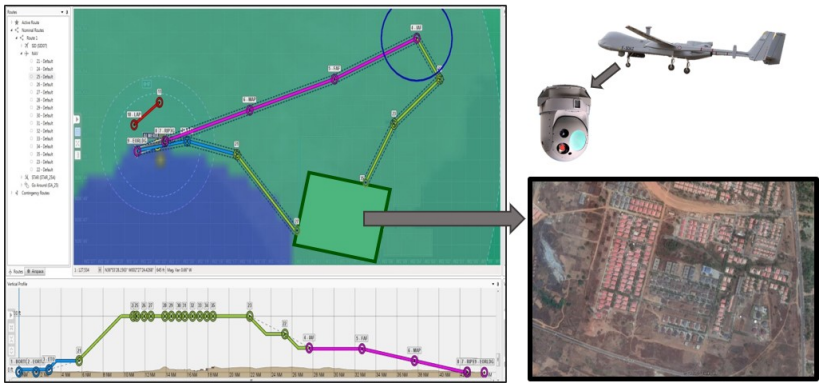


Figure 1.1 Airbus GCS mission planner interface with a square search mission where a UAV equipped with a camera has to look for a target with an unknown location.

a target with uncertain location has to be decided and this thesis aims to solve this problem.

Therefore, the general objective of the thesis is designing algorithms that propose optimized search routes considering the available uncertain information about the search scenario. More specific objectives of the thesis are described below.

- Reviewing the state of the art of probabilistic search in general and of the MTS problem in particular.
- Proposing MTS algorithms that are able to deal with the problem intrinsic complexity and find a good balance between the quality of the solutions and the computational time required for obtaining them.
- Proposing new MTS algorithms that consider realistic aspects of the problem such as UAV motion models that consider the dynamic restrictions of fixed-wing UAVs and realistic sensor models.
- Analyzing the requirements and developing the necessary interfaces in order to allow the use of the thesis contributions by Airbus Research & Technology (R&T) GCS (which is a ground control station developed by Airbus) and a open-source GCS based on QgroundControl (developed during SAVIER project).

1.3. Approach

All planning problems and control problems can be formulated as an optimization problem that aims to minimize (or maximize) a criterion choosing the right actions. More concretely, *Planning* can be defined as the task of finding a sequence of actions that will achieve a goal (Russell and Norvig, 2016). Classical planning only considers deterministic and static environments. However, to be able to deal with real world environments and non-perfect sensors, the planner has to deal with incomplete and uncertain information in a robust manner (Skoglar, 2007).

In this regard, MTS can be expressed as a planning problem that aims to minimize the time of target detection and can be formulated in a probabilistic way in order to deal with the uncertain information inherent to the problem (related to the sensors performance and the target location and dynamics).

This thesis tackles the MTS problem from a new approach based on Ant Colony Optimization (ACO) techniques. We choose this metaheuristic due to its successful application in a variety of problems and because we think that the possibility offered by ACO of including specific knowledge about the problem is an interesting option for a problem like MTS, where a good balance between the quality of solutions and the computational time required to obtain them should be found.

1.4. Main Thesis Contributions

This section summarizes the key contributions and the methodology followed during the thesis.

Literature review. The research of this thesis started with a study of the existing methods that optimize the search routes of a fleet of UAVs in uncertain environments, focusing on the ones that deal with MTS, and highlighting the advantages and limitations of the different state of the art approaches.

The review of the state of the art is contained in Chapter 2.

MTS algorithms based on ant colony based techniques. This thesis presents a new approach based on ant colony optimization to determine the trajectories of a fleet of unmanned air vehicles looking for a lost target in the minimum possible time. We propose two ant colony based algorithms based on ACO techniques for discrete optimization problems that exploit the knowledge of a new MTS heuristic to quickly obtain high-quality UAVs search trajectories. These algorithms propose high-level search trajectories, which are more suitable for rotatory-wing UAVs.

This approach was first presented in the following national congress publication:

- Sara Pérez Carabaza, Eva Besada Portas, José Antonio López Orozco and Jesús Manuel de la Cruz, *Resolución del problema de búsqueda en tiempo mínimo mediante colonias de hormigas*, Actas XXXVI Jornadas de Automática, Bilbao 2015.

Later, it was extended with a deeper analysis of the algorithms and a comparison with several state of the art methods and published in the following journal publication:

- Sara Pérez Carabaza, Eva Besada Portas, José Antonio López Orozco and Jesús Manuel de la Cruz, *Ant colony optimization for multi-UAV minimum time search in uncertain domains*, Journal of Applied Soft Computing, 2018.

MTS algorithms with realistic models. We propose two MTS algorithms for a complex formulation of MTS problem, which optimize multiple criteria and considers realistic UAV models. On one side, apart from optimizing a MTS related criteria this approach avoids the collisions between the UAVs and prevents the UAVs from overflying predefined forbidden regions of the search area. On the other, we consider a radar detection function that presents a realistic behavior and a parametrizable continuous UAV dynamic model suitable for fixed-wing UAVs. To deal with this added complexity the optimization problem in sequentially optimized within a receding horizon controller approach. Besides, as this approach may in turn result in myopic (local) solutions, we improve the quality of the solutions by the optimization of a new myopia avoidance criterion.

In order to solve this problem we first proposed a MTS algorithm based on genetic algorithms, a widely known metaheuristic that has been successfully applied to a variety of application domains. This work was presented at the Genetic and Evolutionary Computation Conference and nominated for best paper of the Real World Applications track:

- Sara Pérez Carabaza, Eva Besada Portas, José Antonio López Orozco and Jesús Manuel de la Cruz, *A real world multi-UAV evolutionary planner for minimum time target detection*, Proceedings of Genetic and Evolutionary Computation Conference (GECCO), Denver 2016.

Besides, encouraged for the good performance on ant colony techniques in the discrete domain, we proposed a MTS algorithm based on an ant colony metaheuristic for continuous domain optimization problems. In order to benefit from MTS specific knowledge, the algorithm incorporates a percentage of ants that use the information of a new MTS heuristic to construct their path. This work was presented also at the Genetic and Evolutionary Computation Conference:

- Sara Pérez Carabaza, Julián Bermudez Ortega, Eva Besada Portas, José Antonio López Orozco and Jesús Manuel de la Cruz, *A multi-UAV minimum time search planner based on ACOR*, Proceedings of Genetic and Evolutionary Computation Conference (GECCO), Berlín 2017.

Integration with a Ground Control Station. Finally, we have integrated the developed contributions within two Ground Control Station (GCS). To this end, we have first analyzed the main characteristics that the MTS algorithms or planners should have in order to facilitate their use for a GCS operator. Considering these requirements, we have developed a prototype that allows operator/user to define the search scenario, to optimize the UAVs routes by means of a MTS algorithm, to analyze the proposed search trajectories and to establish connection with two different GCS. On the one hand, the integration with Airbus R&T GCS, which was one of the main

initial objectives of the thesis as part of SAVIER project, allows to test the thesis contributions following NATO (North Atlantic Treaty Organization) communication standards with a complex UAS developed and used by Airbus. On the other hand, the integration with a second GCS based on the open-source QGroundControl station allows to test the thesis contributions for multi-UAV missions and to prove the generality of the developed planner.

The integration process with Airbus R&T GCS was described in the following national congress paper:

- Sara Pérez Carabaza, Eva Besada Portas, José Antonio López Orozco and Jesús Manuel de la Cruz, *Planificador de búsqueda en tiempo mínimo en un sistema de control de RPAS*, Actas XXXVII Jornadas de Automática, Madrid 2016.

In order to integrate the contributions of this thesis with a GCS, a methodology that enables the automatization of the definition of the target probability models was identified by Airbus as a requirement. The proposed methodology is described in Chapter 6 and in the article listed below, published in the Journal of Sensors. This article also presents a MTS approach based on genetic algorithms for the simultaneous optimization of the control commands of the UAVs and of the cameras orientation, whose probabilistic camera model considers the terrain elevation. Examples of the results obtained by this approach during the integration process with Airbus R&T GCS are presented in Chapter 6, although a complete description of the approach is not covered in this thesis.

- Sara Pérez Carabaza, Eva Besada Portas, José Antonio López Orozco and Gonzalo Pajares, *Minimum time search in real-world scenarios using multiple UAVs with onboard orientable cameras*, Journal of Sensors, 2019.

1.5. Thesis Organization

The thesis is structured as follows:

- **Chapter 1** introduces the problem (Minimum Time Search, MTS) solved in this thesis, states its main objectives and outlines the approach followed to tackle them.
- **Chapter 2** reviews the state of the art of MTS. First, it relates MTS with other similar problems of the literature, outlining their similarities and differences. Then, it provides a historical background of the problem and reviews the state of art that has motivated the thesis in more detail.
- **Chapter 3** begins by stating mathematically the MTS objective. Then, it presents how the uncertainty sources of information inherent to MTS (and to probabilistic search problems in general) are modeled and updated with new information through a Recursive Bayesian Filter (RBF). Next, it formulates the most common fitness criteria strategies followed in Probabilistic Search (PS) and justifies the selection of the Expected Time (ET) of target detection for MTS. Finally, the chapter describes from a general point of view the approach followed by PS algorithms to solve the search problem and explains in more detail the main characteristics of the approach followed by this thesis, consisting in the use of ant colony based techniques in combination with problem specific heuristic information.
- **Chapter 4** considers a simplified formulation of the MTS problem, which considers UAV models appropriate for rotatory-wing UAVs, and proposes and analyzes several MTS algorithms based on discrete optimization techniques. The main objective of the chapter is testing the power of ant colony techniques and the consideration of using ad-hoc heuristic for MTS. It starts describing the selected discrete UAV models, the codification of the solutions (search trajectories as sequences of adjacent cells of the discretized search area) and the evaluation

criterion. Then, it presents the proposed MTS algorithms based on a discrete ant colony based technique that benefits from MTS heuristic information.

- **Chapter 5** focuses on a more complex formulation of the MTS problem, which considers UAV models appropriate for fixed-wing UAVs, and which is solved employing continuous optimization techniques. The main objective of the chapter is testing if the ant based approach with ad-hoc heuristic information is also beneficial in a more realistic version of the problem. Following the same structure of Chapter 4, the chapter starts by describing the selected UAV models, the codification of the solutions and the developed multi-criteria and receding horizon approach. Next, the chapter presents the proposed MTS algorithm based on a continuous ant colony based technique, which includes heuristic information through the inclusion of specialized heuristic ants. Besides, we compare its performance with a MTS algorithm based on genetic algorithms developed during this thesis.

- **Chapter 6** summarizes the work done under the SAVIER project with the objective of integrating the capabilities of the MTS algorithms within two different Ground Control Stations (GCS). After a brief introduction to SAVIER project, the chapter describes the architecture followed for the integration, the requirements that have been identified to enable the use of MTS algorithms for different GCS and the developed Graphical User Interface (GUI). Finally, the specifications and results of both integration processes are explained in more detail.

- **Chapter 7** summarizes the main conclusions of the thesis and proposes possible future research lines.

Finally, it is worth noting that this thesis aims to be useful for anyone interested in trajectory optimization in uncertain environments or in solving optimization problems considering problem specific heuristic information through the use of ant colony based algorithms. With this purpose in mind, each chapter is mostly self-contained to facilitate the understanding of the information of interest to each reader.

Chapter 2

State of the Art

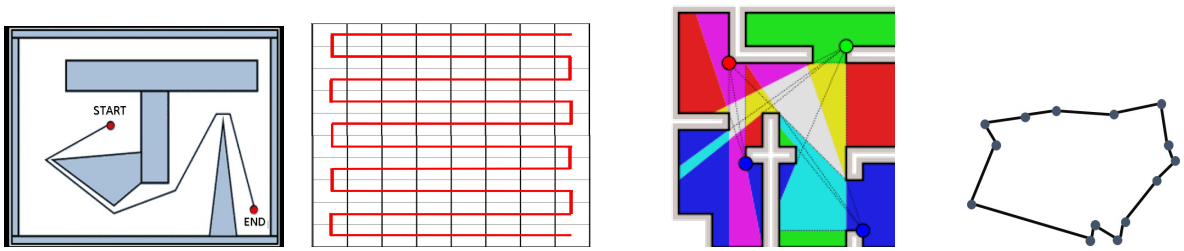
"If I have seen further it is by standing on the shoulders of giants."

Isaac Newton

This chapter discusses the state of the art of Minimum Time Search (MTS) problem, analysing with greater detail several works that have motivated this thesis. The chapter is divided into two sections. The first one discusses, from a general point of view, related probabilistic search problems such as coverage or the Travelling Salesman Problem (TSP), stressing their common characteristics and differences with MTS. The second section analyzes in more detail the state of the art of Probabilistic Search (PS), which aims to find the best Unmanned Vehicles (UV) search trajectories in uncertain environments and which encompasses the MTS problem.

2.1. Minimum Time Search and Related Problems

The wide research field of robot motion planning is closely related with probabilistic search and is especially interesting because many of the techniques and models used to solve those problems can be adapted to solve PS. The objective of path planning problems is to optimize a feasible route of a vehicle from its initial location to a goal destination. Multi-vehicle extensions are very common too and consist on



(a) Motion Planning (b) Coverage (c) Art Gallery Problem (d) TSP

Figure 2.1 Related MTS problems. a) Path avoiding obstacles (blue polygons). b) Coverage zigzag pattern (red line). c) Sensors (colored circles) and their areas of visibility in different colors. d) TSP trajectory along all the cities (blue circles).

optimizing the route of a fleet of robots (Raja and Pugazhenthi, 2012). Depending on the problem, the robot may have to fulfill some constraints, for example to avoid obstacles (as the example of Figure 2.1 (a) shows) or dangerous locations (Jun and D'Andrea, 2003). The optimization objective also depends on the problem, although many works optimize the travel cost (time or fuel). In both PS and motion planning, the trajectories of one or several vehicles are optimized, but in PS the optimization is done according to some criteria related to the environment gathered information and it takes advantage of uncertain target location information. Moreover, while in motion planning the time of the mission is not fixed, typically in PS problems there are limited resources and the trajectories are optimized up to a fixed time (that can be determined for example by the type of mission or fuel). Finally, in both problems the initial vehicle states (e.g. locations) are fixed, but in PS typically the final points are not given, as the key objective is related to wisely explore the environment and not to achieve a destination.

Another large class of related problems is target tracking, which corresponds to the task that arises after one or several targets have been detected or assigned (Pulford, 2005). The goal of target tracking is to maintain over time the targets within the trackers (robots) sensor ranges. Tracking and PS are both target related problems but still have obvious differences. While in PS the objective is to optimize the route for finding the target, the tracking objective aims to avoid losing the knowledge about its position afterwards. These problems occur sequentially and are generally solved

with different approaches, therefore it is worth mentioning the work presented in (Furukawa et al., 2006), where both search and tracking are treated jointly. A survey and taxonomy of search and tracking can be found in (Robin and Lacroix, 2016).

Another closely related problem to path planning is coverage, whose main objective is to determine the path of a robot in order to explore a whole area and it has applications such as robotic demining or lawn mowing (Fan and Jin, 2010). Typically used coverage methods are predefined patterns like spiral or zig-zag (as the example of Figure 2.1(b) shows), but variations/combinations or other methods are required for complex scenarios (e.g. where several vehicles are involved or there are areas with obstacles). In both PS and coverage one or several vehicles have on-board sensors that allow them to explore an area of interest. However, while in coverage the important objective is to explore the whole area efficiently without a mission time limit, in PS the mission time is usually limited and therefore is not possible to fully explore the area. Besides, while in coverage planning all areas have the same importance, usually in PS some areas are crucial to be explored during the limited mission time. Moreover, specifically in Minimum Time Search (MTS), not only it is important to visit the areas with high probabilities of target presence, but also it is necessary to visit them soon and in the best order to reduce the target detection time. Nevertheless, if there is no initial information available about the target location (and hence a uniform belief over the search area is the best probability distribution to describe the initial target location) and the target is static, the PS problem can be seen as a coverage problem with limited resources and coverage methods can successfully solve it. For this reason, in order to take advantage of PS methods is important to have an informative prior belief and/or a target motion model.

Another beautiful problem which is also related to coverage is the art gallery problem (illustrated in Figure 2.1(c)). The problem, also categorized as static surveillance by (Robin and Lacroix, 2016), consists in determining the minimum number of sensors (security guards) and to allocate them to be able to observe properly the whole gallery. Analogously to the coverage problem, the objective is to cover the scenario

with several sensors, however in contrast to coverage and PS problems, the sensors deployed in the art gallery problem are static.

The widely known Travelling Salesman Problem (TSP) consists in determining the shortest closed loop that traverses once a group of cities (a solution example is displayed in Figure 2.1 (d)). Besides, several variations like the employment of multiple salesmen can be found in the literature (Goyal, 2010). Although the objective of both TSP and PS is to efficiently visit several locations (cities and areas respectively), two main characteristics differentiate both problems. First, while in PS the initial positions of the searchers are fixed (often determined by the entry point of the vehicle into the area), in TSP there is no initial position (as the solution is a closed loop). Second, while in TSP there is no maximum time restriction, in PS the mission time is generally limited. And lastly, as we have already mentioned, the study of related problems is interesting because the methods applied to solve them can often be adapted to the problem at hand. This is the case of this thesis, which proposes to apply ant colony based methods, widely known for solving TSP, to MTS.

Finally, within the search problems, we can distinguish two groups according to the target intentions: one-sided search and two-sided search. While in the first group the targets movements are independent of the searchers actions, in the second group the targets react to the searchers movements. Two-sided search literature mainly focuses on search games or adversarial search, where the targets try to avoid being detected. One-sided search literature is wider and includes the works analyzed in the following sections and the methods presented in this thesis. An extended survey of both one-sided and two-sided search works can be found in (Chung et al., 2011).

2.2. Probabilistic Search

The works analyzed in this section have a close relation with the ones developed during this thesis, as all of them address cooperative searching problems in uncertain environments with limited resources, and to do it, they make the most of the prior

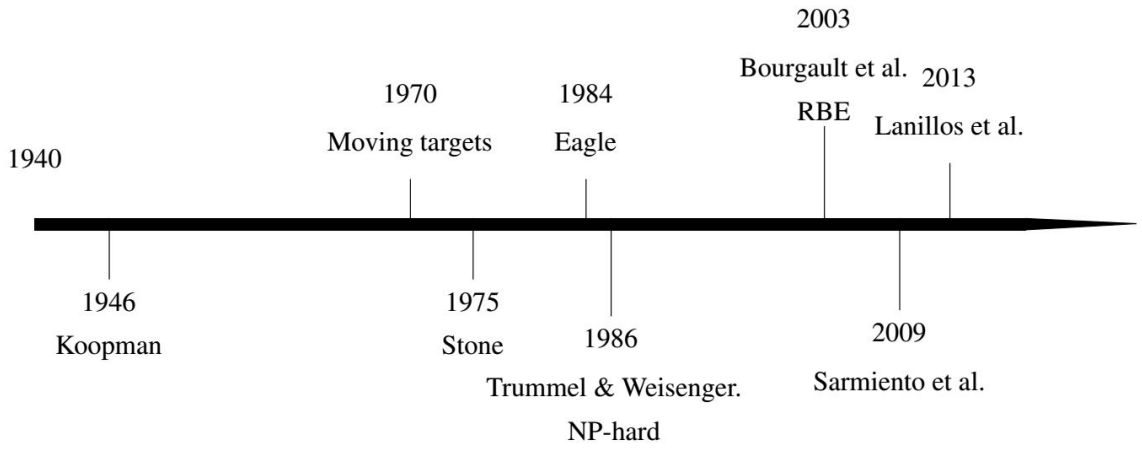


Figure 2.2 Timeline with the most relevant works for PS from 1940s until nowadays.

information (relative to the target location and dynamics, the sensor performance and the environment).

This section starts with an overview of the historical background of the first works in search theory. Then, several relevant closely related works are analyzed and compared, attending to different characteristics relative to the target, the autonomous vehicles that carry out the search, the environment and the algorithms and techniques employed to solve the problems.

2.2.1. Historical Background

The timeline of Figure 2.2 summarizes the most relevant works for PS since the beginning of search theory until nowadays. Search theory had its beginnings in the naval operations research done by the U.S. Navy's anti-submarine research group (ASWORG) during World War II. The techniques developed during this period (1942-1945) were summarized in Koopman's report "Search and Screening" (Koopman, 1946). The report, originally confidential and later updated in (Koopman, 1980), sets the foundations of search theory. However, the initially developed theory has two important drawbacks: it assumed that the space was infinitely divisible (i.e. their approach did not consider the searchers dynamics or spatial restrictions to move from one location to another) and considered non-realistic assumptions for

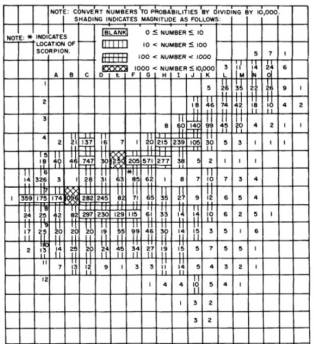
all types of sensors (e.g. the sensors were always able to detect a target within their measuring range).

It took until 1960s to see the first application of search theory, when it had a crucial role during the search for the four hydrogen bombs lost after a plane crashed near Palomares (Almería, Spain) in 1966 and during the search for the USS Scorpion submarine lost in 1968 somewhere between the 4000 km that separates Azores islands (in the Atlantic ocean) from Norfolk (United States). In this early stage of search theory, the success of the methods was due to the building of the probability map and the modelling of the sensor performance. During the search for the four nuclear bombs near Palomares, three of them were found within the first day, but as the search for the fourth one was still unsuccessful after several days, the experts decided to apply bayesian theory in order to obtain a probability map with the most promising locations to search (McGrayne, 2014). The probability map was constructed considering the probabilities of the different possible scenarios given by experts (e.g. fail of the bomb parachutes) and the information provided by a witness, and thanks to the use of the probability map the fourth bomb, displayed in Figure 2.3 (a), was finally found. As the group in charge of finding the USS Scorpion submarine was the same one that found the bomb in Palomares, they decided to use again the same successful bayesian method to construct the prior submarine location belief. In this case, the probability map was constructed based on nine different possible scenarios with associated credibility weights (Richardson and Stone, 1971). Remarkably, the submarine was found at 240 meters of the cell with highest probability. Figure 2.3 (b) shows a photo of the submarine and Figure 2.3 (c) displays the initial probability map used for its search.

In 1975, Lawrence Stone, a mathematician whose work in the USS Scorpion search help him to become an expert in search theory, wrote his classic book *Theory of Optimal Search* (Stone, 1975). The book, which mainly focuses in static targets and was awarded with Lanchester Prize, had a high influence in posteriori works. Some years later, the naval engineer Eagle observed that in case that some dynamical restrictions were considered in the search path, the methods proposed by Stone did

(a) Founded fourth H-bomb

(b) USS Scorpion



(c) USS Scorpion prior belief

Figure 2.3 Search for lost nuclear bomb and USS submarine during 1960s.

not work any more. Under this new assumption, the search path of a unique searcher with dynamic restrictions (only allowed to move to neighbor cells) was solved using Partially Observable Markov Decision Process (POMDP) methods for a simple scenario (a grid of three by three cells). The methods that appeared after this work have been more oriented towards maximizing the probability of target detection than towards minimizing the target detection time.

The first works in search theory that consider a moving target appeared in 1970s. At the beginning, these works dealt with very simple scenarios that consider only two cells, as happens in (Pollock, 1970) or (Dobbie, 1974). The research of moving targets has been extended to bigger scenarios and continues nowadays.

The increasing availability of cheap and powerful computers that started around 1975 had an important impact in search theory and brought attention of more researchers into the field. Works shifted from previous mathematical approaches and analytic solutions to algorithmic ones that could deal with more complex and realistic scenarios. The solutions algorithmically obtained were no longer optimal, as it was understood that due the high complexity of the problem it was only possible to find optimal solutions of naive and simple instances. Moreover, the complexity of the constraint search problem for a given planning horizon and with a discrete time and space formulation was characterized to be NP-hard (Trummel and Weisinger, 1986).

With the widespread use of drones from 2000s until now, Unmanned Aerial Vehicles (UAVs) mission optimization problems have attracted increasing number of researchers (Fuhrmann and Horowitz, 2017). The reduced cost of the UAVs and higher capabilities of computers make possible the optimization of search paths of several UAVs with reasonable financial and time resources. Among the most recent search theory works, it is worth mentioning the article by Bourgault, Furukawa and Durrant-Whyte (Bourgault et al., 2006) for introducing a general Bayesian framework for the searching problem that has been widely adopted in most of the works (including this thesis) since its publication. The proposed bayesian filter allows to update the target belief with the target movements and sensor measurements and is particularly suitable for combining, in a rational manner, non-linear motion models and heterogeneous non-gaussian sensor measurements with other sources of quantitative and qualitative information.

Finally, we want to make emphasis in the works that deal specifically with MTS. Although the importance of minimizing the search time is mentioned from the beginning and optimized in some early works (Pollock, 1970), most of the works that can be found in the literature maximize the probability of detection. The work presented in (Bourgault et al., 2006) states both the expected target detection time and probability of detection strategies, but only optimizes the second one in the simulations included in its result section. The work done in (Sarmiento et al., 2009) optimizes the expected time of detecting a static target by a unique searcher that can move in a structured environment (e.g. building). Lastly, it is also important to mention the recent work by Lanillos et al. in the MTS problem, applying several methods like cross entropy optimization (Lanillos et al., 2012) or bayesian optimization algorithm (Lanillos et al., 2013) in order to optimize the expected detection time of a static or a moving target. All these works are analyzed in more detail in the following sections.

2.2.2. Literature Review

In this section the approaches of recent works that tackle probabilistic search problems are analyzed and compared. This literature overview is not exhaustive and only the works more relevant for this thesis are analyzed. Other surveys can be found in (Benkoski et al., 1991), (Chung et al., 2011) and (Robin and Lacroix, 2016).

As numerous properties differentiate some PS approaches from others, the following comparison has been organized attending to the characteristics related to the different elements of the search: the target, the UAVs, the environment and the employed algorithm. Moreover, the comparison is summarized in several tables below, where each row highlights the characteristics of a different work (including the ones presented in this thesis). Finally, the works are organized by publishing date, leaving the main contributions proposed in Chapters 4 and 5 of this thesis to the last two rows.

2.2.2.1. Target

Table 2.1 contains an analysis of the most relevant works according to four categories related to the targets under search: information modelling approach, restriction of the prior belief distributions and whether they consider static/dynamic or single/multiple targets. Each row of Table 2.1 corresponds to a different work and their categorization can be seen from the second to the fifth column.

As the target location is unknown, it is modelled as a random variable and its uncertainty described with a Probability Density Function (PDF). The knowledge about the target location contained in the PDF can be updated with the information obtained from the sensor measurements and target dynamic information through a bayesian filter. In order to compute and update the PDF, the PDF has to be defined using a *probabilistic modelling approach*. The ones used in the works under analysis are indicated in the second column of Table 2.1. Most of the works and this thesis employ a space discretization of the PDF into a regular grid (labelled as probability

Table 2.1 Search works comparison according to the target.

Work	Information Modelling	Specific Beliefs	Moving Target	Multiple Targets
(Eagle, 1984)	probability map		✓	
(Yang et al., 2002)	probability map			✓
(Wong et al., 2005)	probability maps		✓	✓
(Bourgault et al., 2006)	probability map		✓	
(Sarmiento et al., 2009)	PDF	uniform		
(Hoffmann et al., 2006)	particle filter			
(Lin and Goodrich, 2009)	probability map			
(Tisdale et al., 2009)	probability map			
(Gan and Sukkarieh, 2010)	probability map			
(Delle Fave et al., 2010)	probability map		✓	
(Lanillos et al., 2012)	probability map		✓	
(Lo et al., 2012)	probability map			
(Lanillos et al., 2013)	probability map		✓	
(Carpin et al., 2013)	probabilistic map (quadtree)			
(Lanillos et al., 2014a)	probability map		✓	
(Lanillos et al., 2014b)	probability maps			✓
(Berger et al., 2014)	probability maps			✓
(Khan et al., 2015)	probability map	uniform		✓
(Chang-jian et al., 2015)	probability map		✓	✓
(Berger et al., 2016)	probability maps			
(Meghjani et al., 2016)	probability maps	gaussian		✓
(Raap et al., 2016)	probabilistic map (hexagonal cells)		✓	
(Pérez-Carabaza et al., 2016b)	probability map			
(Yao et al., 2017)	probability map			
(Pérez-Carabaza et al., 2017)	probability map		✓	
(San Juan et al., 2018)	probability map			✓
(Pérez-Carabaza et al., 2018)	probability map		✓	
Chapter 4	probability map		✓	
Chapter 5	probability map		✓	

map in Table 2.1), where the probability of each cell equals the integral value of the PDF between its space limits. All the works that use grid-based PDFs consider a grid of square cells of fixed size with the exceptions of (Raap et al., 2016) and (Carpin et al., 2013). The former considers hexagonal cells, which have the advantage of having the same distance from neighbor cells. The latter uses a quadtree representation

of the belief to represent a variable resolution grid-based belief. Other possibility, employed by (Hoffmann et al., 2006), is to maintain an estimate of the target state's PDF using a particle filter. (Sarmiento et al., 2009) is a singular case, where the specific characteristics of the PS problem (initial uniform belief, static target and properties of the sensor) allow to consider the belief as a generic PDF without any discretization, update its value and to compute easily the objective function.

In the third column of Table 2.1 we can observe that most of the analyzed works and this thesis do not impose any requirement for the type of belief. Hence, they are not designed for *specific beliefs* and can be applied to different kinds of initial beliefs (gaussians, multi-modal gaussians, uniform, etc). On the contrary, the methods proposed by (Sarmiento et al., 2009) and (Khan et al., 2015) are particularly designed for scenarios where there is no initial knowledge about the target location (uniform belief). (Meghjani et al., 2016) does not impose restrictions for the belief of single target scenarios, but in the case of multiple targets the belief of each target has to be described by a two dimensional Gaussian function (centered in the most probable location of each target).

Besides, we can organize the works attending whether they allow to include a target dynamical model or not. Works that do allow the inclusion of a target dynamical model are indicated with a tick in the *Moving Target* column of Table 2.1. Moreover, most of these works consider Markov motion models that allow to predict the target belief at time step t from the previous target belief at the time step $t-1$. More concretely, (Delle Fave et al., 2010) considers a simple Markov motion model where the target moves randomly to adjacent cells. (Lanillos et al., 2013) and (Lanillos et al., 2014a) use different target motions models, some based on sea currents, and (Lanillos et al., 2012) tests the performance of the proposed algorithm over a scenario whose target movement model is based on a wind database. Besides, in (Bourgault et al., 2006) a life-boat moved by sea currents is searched. Moreover, in (Chang-jian et al., 2015), the target velocity is changed along time. Finally, (Raap et al., 2016) is a special case where the target position is unknown but the target dynamics is defined by a deterministic motion model known by the PS algorithm before the search starts.

A recent and extensive review of probabilistic search algorithms for moving targets can be found in (Raap et al., 2019).

The last category presented in last column of Table 2.1 indicates whether the works search for a unique target or for *multiple targets*. In single target grid-based beliefs, each cell value indicates the probability of the target existence within the cell. Hence, assuming that the target is inside the search area, the sum of the probabilities of target presence of all the cells equals one. Between the multi target works, we can distinguish two approaches: works that consider that all the targets follow the same probability distribution (homogeneous targets) and works that maintain different distributions for each target (heterogeneous targets). Within the first group, (Chang-jian et al., 2015; Khan et al., 2015; Yang et al., 2002) and (San Juan et al., 2018) model the location information of an unknown number of targets with an unique probability distribution, where each cell value indicates the probability of target existence. Thus, in this case the sum of the target presence probabilities of all the cells of the probability map can be bigger than one. Within the second group, (Berger et al., 2014; Wong et al., 2005) and (Meghjani et al., 2016) maintain a separate belief for a known number of targets. Hence, these works maintain multiple *probability maps* as it is indicated in the information modelling column. Furthermore, (Lanillos et al., 2014b) implement both approaches: in their separated approach there is one belief for each target and in their unified approach a common belief obtained from the union of the beliefs of each target is used.

Finally, it is worth noting that this thesis and its associated works published in (Pérez-Carabaza et al., 2017) and (Pérez-Carabaza et al., 2018) propose search methods based on ant colony optimization for both static or dynamic targets, whose location information is modelled with a probability map and its target dynamic information is captured with a Markov dynamical model. Besides, the type of the probability maps is not restricted. On the contrary, we have considered a large variety of beliefs and target dynamical models that allow to analyze the general performance of the algorithms over a variety of scenarios. Furthermore, it is worth mentioning that when it can be assumed that the targets follow the same distribution, the methods for one

target (such as the ones proposed in this thesis) can be straightforward applied to the search for multiple targets. In that case, the search does not finish when one target is found, but it continues until all targets are found (Lanillos et al., 2012). However, adapting single target algorithms to multiple independent targets with different beliefs is not trivial. Hence, the search for heterogeneous targets, which do not follow the same location distribution or that have different dynamics, is out of scope of the thesis.

2.2.2.2. Unmanned System (US)

The search is made by unmanned systems that carry sensors that enable the detection of the target. All the analyzed works and this thesis use unmanned aerial vehicles (UAVs) for the search, with the exception of (Sarmiento et al., 2009) that optimizes the route of an unmanned ground vehicle. The works have been classified in Table 2.2 according to four different categories related to their Unmanned System (US): single vs multiple vehicle formulations, sensor models performance and range, and US dynamical models.

Within the first category, summarized in the second column of Table 2.2, a great part of the state-of-the-art algorithms allow to use *multiple vehicles*. Although this has a great advantage over the search results, since more areas can be covered in the same amount of time, it increases the solution space and the complexity of the problem. In contrast with other multi-vehicle problems, most of the multi-vehicle search works do not require an additional cooperation strategy, as the cooperation between the UAVs arises naturally from the optimization of the fitness criteria. For instance, for obtaining a high probability of detection or a low expected detection time in scenarios with static targets and whose belief is spread over several high probability areas, typically the US need to distribute their forces along the search area, as there is not usually much gain in this type of scenario when a vehicle explores an area that it has already been efficiently explored by another vehicle. On the contrary, several vehicles may focus on the same area if there is a reward: for example due to the limited sensor capabilities the exploration of the same area by several vehicles may

Table 2.2 Search works comparison according to the UAVs.

Work	Multi-US	Sensor Performance	Sensor Range	Dynamic Model
(Eagle, 1984)		ideal	cell	discrete
(Yang et al., 2002)	✓	ideal	cell	discrete
(Wong et al., 2005)	✓	FN	wide range	continuous
(Bourgault et al., 2006)		FN	wide range	continuous
(Hoffmann et al., 2006)	✓	FN	limited footprint	continuous
(Sarmiento et al., 2009)		ideal	wide range	discrete/cont.
(Lin and Goodrich, 2009)		ideal	cell	discrete
(Tisdale et al., 2009)	✓	FN	limited footprint	continuous
(Gan and Sukkarieh, 2010)	✓	FN	wide range	continuous
(Delle Fave et al., 2010)	✓	FN	limited footprint	continuous
(Lanillos et al., 2012)		ideal	cell	discrete
(Lo et al., 2012)	✓	FN	cell	discrete
(Lanillos et al., 2013)	✓	ideal	cell	discrete
(Carpin et al., 2013)		FN/FP	cell	waypoints
(Lanillos et al., 2014a)	✓	FN/FP	wide range	discrete
(Lanillos et al., 2014b)	✓	FN	wide range	continuous
(Berger et al., 2014)	✓	FN	cell	discrete
(Khan et al., 2015)	✓	FN/FP	cell	discrete
(Chang-jian et al., 2015)	✓	FN	cell	discrete
(Berger et al., 2016)	✓	FN	cell	discrete
(Meghjani et al., 2016)		ideal	cell	discrete
(Raap et al., 2016)		ideal	limited footprint	discrete
(Pérez-Carabaza et al., 2016b)	✓	FN/FP	wide range	continuous
(Yao et al., 2017)	✓	FN	limited footprint	discrete
(Pérez-Carabaza et al., 2017)	✓	FN/FP	wide range	continuous
(San Juan et al., 2018)	✓	ideal	cell/ lim. footprint	discrete
(Pérez-Carabaza et al., 2018)	✓	ideal	cell	discrete
Chapter 4	✓	ideal	cell	discrete
Chapter 5	✓	FN	wide range	continuous

be beneficial in some types of scenarios. Hence, the US are expected to distribute themselves or concentrate according to the development of the search mission and the evolution of the target belief.

Second, we have classified in the third column of Table 2.2 the *sensors performance* in three categories: *ideal*, and sensors with False Negative (*FN*) and False Positive (*FP*) rates. Several works are only suitable to sensors with *ideal* performance, i.e. they do not consider the possibility of false negative (also called miss detections) or false positive (also known as false alarms) measurements. However, in reality, sensors do not have ideal performance, as they may fail to detect a target that is present (a false negative) or return a detection when a target is not present (a false positive). The *FN* and *FP* rates will depend on the weather conditions, the sensor and target characteristics, etc. Among the works that do not contemplate ideal performance the majority include the consideration of false negatives (*FN*). Finally, (Carpin et al., 2013; Khan et al., 2015; Lanillos et al., 2014a) and (Chang-jian et al., 2015) consider the possibility of both false positives and false negatives (*FN/FP*).

Third, we have also classified the sensors models according to their visibility *range* in the fourth column of Table 2.2. The more simplified models consider that the sensor footprint (range) coincides with one cell of the map. This assumption is taken in several works (classified with the *cell* label in Table 2.2) and although it can be reasonable in many cases, it forces the belief resolution to be equal to the sensor footprint. Between the works whose sensors cover more than a unique cell, we have distinguished between limited footprint sensors and wide range sensors. The first group is classified as *limited footprint* in Table 2.2 and includes sensors whose footprint has clear limits, typically used to model electro-optic sensors or cameras. Within this group, (Raap et al., 2016) considers ideal discrimination, assuming target detection if the target is inside the footprint; (Tisdale et al., 2009) and (Yao et al., 2017) consider imperfect detection probability when the target is inside the footprint; and (Carpin et al., 2013) assume that the detection probability depends on the UAV height. The final work of this group, presented in (Delle Fave et al., 2010), considers the border effect with a worse detection probability value in the cells in the border of the footprint. The second group is classified as *wide range* in Table 2.2 and includes sensors whose footprint does not have clear limits. The generic wide range model used in (Gan and Sukkarieh, 2010) and (Lanillos et al., 2014b) consider a decreasing

probability of detection with the distance from the sensor to the target. This type of behavior is adequate for modeling ultrasonic sensors (Lanillos et al. (2014a)) or radars ((Bourgault et al., 2006; Lanillos et al., 2014a; Pérez-Carabaza et al., 2017, 2016b; Wong et al., 2005)). More concretely, the radar model used in (Lanillos et al., 2014a; Pérez-Carabaza et al., 2017, 2016b) and in Chapter 5 of this thesis considers a decreasing probability of detection with the distance from the sensor to the target, whose maximum value depends on the different features of the radar, target and the environment. Lastly, the sensor model used in (Sarmiento et al., 2009) is a special case where the sensor of the ground vehicle that moves in a indoor environment can see all the room that is visible from its position.

Finally, the classification regarding the vehicle *dynamical models* used in PS algorithms is especially relevant as it conditions the solution space and determines the shapes of the search trajectories. PS works generally use discrete time dynamical models $s_u^{t+1} = f(s_u^t, c_u^t)$ that given the current state s_u^t of the $u-$th vehicle at time instant t and the control action c_u^t, return the new vehicle state s_u^{t+1}. The use of these models allows the optimization algorithms to directly manipulate the solutions in the control action space (instead of in the trajectory space) and use the UAVs dynamic models to obtain the corresponding UAV trajectories required for evaluation purposes. In the fifth column of Table 2.2 we have divided the dynamical models in three categories: *discrete*, *waypoints* and *continuous*. Under the *discrete* category we include models whose control action domain is discrete and which lead the UAVs flying over a cell of the probability map to one of its neighbor cells. Within this group, (Lanillos et al., 2014a, 2012, 2013) and (Pérez-Carabaza et al., 2018) allow the movements of the UAVs from their current cell to the eight neighbor cells of the square grid following the cardinal directions. (Berger et al., 2016, 2014; Chang-jian et al., 2015; Lo et al., 2012; Yang et al., 2002) and (Yao et al., 2017) also consider movements in the 8 cardinal directions but impose a maximum turn restriction of 45 degrees (allowing the UAVs to choose only between three actions; turn left, continue straight or turn right). Furthermore, (Eagle, 1984) and (Khan et al., 2015) only allow movements in the main four cardinal directions (North, East, South and West). And

finally, (Raap et al., 2016) consider an hexagonal grid where rotatory-wing UAVs can move towards all the neighbor cells (with 6 possible actions) and where fixed-wing UAVs are imposed a maximum turn restriction of 60 degrees (supported by 3 possible actions). Within the second group (categorized as *waypoints* in the table), the algorithm presented by (Carpin et al., 2013) selects a destination cell (waypoint) and then the UAV is directed to it without considering any dynamic restrictions. Similarly, the strategy followed by (Meghjani et al., 2016) also chooses a destination cell, but the UAV is directed toward it considering the discrete 8 cardinal direction model. Finally, the works categorized as *continuous* consider more complex dynamical models with stricter dynamic restrictions that do not impose the UAVs to move from cell to cell of the grid. Therefore, continuous models are more appropriate for fixed-wing UAVs that need smoother trajectories than rotatory-wing UAVs. Within this group, (Gan and Sukkarieh, 2010) and (Lanillos et al., 2014b) consider simple linear constant velocity models without imposing any limits in the control actions, where the UAV heading is controlled through an instantaneous turn rate command. Besides, (Delle Fave et al., 2010) also employs a linear constant velocity model with a limited bank angle of 25 degrees that conditions the UAVs turn rate, and (Hoffmann et al., 2006) considers a linear model that restricts the turn rate to [-1, 1] degrees per unit time and the velocity to [-4,4] meters per square unit time. Furthermore, (Bourgault et al., 2006) and (Wong et al., 2005) employ a non-linear constant velocity model with a limited turn rate per unit time. Additionally, (Pérez-Carabaza et al., 2016b) and (Pérez-Carabaza et al., 2017) use a non-linear Simulink model with limited bank angle, velocity and height. Finally, (Sarmiento et al., 2009) is also a special case regarding its vehicle motion model, as it optimizes the trajectory of a unique searcher in two levels: it first optimizes the room visiting order of the structured environment and then it optimizes the robot paths considering its dynamic restrictions.

In this thesis, we consider a discrete cardinal UAV dynamical model and an ideal and single cell sensor model in Chapter 4, and a non-linear Simulink UAV dynamical model with restrictions and a radar (wide range) sensor model in Chapter 5. The research in Chapter 4 (mostly contained in (Pérez-Carabaza et al., 2018)) was

done with the main intention of analysing the performance of different optimization techniques and heuristics. The research in Chapter 5 (mostly published in (Pérez-Carabaza et al., 2016b) and (Pérez-Carabaza et al., 2017)) was done with the main purpose of bringing realism to the models. Therefore, the algorithms proposed in Chapter 4 are more appropriate to rotatory-wing UAVs and thus have the advantage of requiring lower computational times. In contrast, the models considered in Chapter 5 are more realistic and appropriate to fixed-wing UAVs.

2.2.2.3. Environment

Search missions can take place in different environments like maritime search (Bourgault et al., 2006) or (Meghjani et al., 2016), WIlderness Search and Rescue (WISaR) missions (Lin and Goodrich, 2009) or search missions inside a building (Sarmiento et al., 2009). Search is typically categorized according to the environment into *structured* and *unstructured* search. Within the first group, the search area has a clear structure that strongly limits the searchers movements, for example when the search for the target is done inside a building. This is the case of (Sarmiento et al., 2009) that optimizes the route of a ground robot inside several rooms of the same floor whose distribution is known beforehand. Further interesting work in structured search can be found in the thesis by (Lau, 2007). The second group includes most of the works analyzed in this section (including the work of this thesis), where the environment in where the search occurs can be understood as an open free space.

Besides, the environment can contain some restricted areas or Non Flying Zones (NFZ) that must not be overflown by the UAVs. This is the case of (Yao et al., 2017) where the search is taken in an environment with several buildings and considers high-rise buildings as non flying zones that the UAVs must avoid. This is also the case in (Pérez-Carabaza et al., 2016b) and (Pérez-Carabaza et al., 2017), where one or several *non-flying zone*s are defined over the grid-based environment.

Moreover, the environment can affect also the performance of the sensors during the search. For this reason, several works consider parametrizable sensors models

whose performance parameters may be affected for example by the weather conditions. Besides, the objects in the scene may occlude some areas to the sensors. This happens for example in the environment considered by (Yao et al., 2017), where the UAVs fly at low altitudes, and the search area contains several tall buildings that can occlude the sensor measurements.

In this thesis the search environment is considered as an open space where the UAVs can move freely with the exception of the NFZ defined over the grid-based environment (Pérez-Carabaza et al., 2017). Besides the sensor models can also be adapted to the environmental conditions and sensor capabilities (Pérez-Carabaza et al., 2017) and (Pérez-Carabaza et al., 2019).

2.2.2.4. Algorithms

After reviewing the state of the art attending to the properties related to the target, the UAVs and the environment, this section finalizes the review highlighting the general properties of the optimization algorithms and techniques used to tackle the PS problems. Table 2.3 shows their fitness criteria, the optimization method used to find the UAV trajectories, as well as the possibilities of using the algorithm offline or online, implemented in a centralized/decentralized fashion, including a myopia reduction mechanism or applying constructive heuristics.

The *objective function* optimized by each planner, presented in the second column of Table 2.3 and explained below, is directly dependent on the problem tackled by each work. Moreover, its value, evaluated for a given set of UAV search trajectories, often depends on the initial target belief and dynamic model, and on the sensor model. The works under analysis optimize the following objective functions:

- Great part of probabilistic search works optimize the probability of finding the target, by maximizing the chances of finding the target/targets at any point along the optimized UAV trajectories. All these works are identified with the P_d label in the second column of Table 2.3. The case of (Gan and Sukkarieh, 2010) is equivalent, but as $1 - P_d$ (which stands for the probability of no detecting the

Table 2.3 Search works comparison according to the algorithms.

Work	Obj. Function	Optimization Method	Offline /Online	Centr. /Distr.	Coll.	Myopia Heur.	Constr. Heur.
(Eagle, 1984)	P_d	POMDP-DP	offline	CEN			
(Yang et al., 2002)	P_d + IG	Q-learning	online	DIS.		✓	
(Wong et al., 2005)	$\sum_{k=1}^{Q} w_k P_d^k$	SQP	offline	DIS		✓	
(Bourgault et al., 2006)	P_d	Greedy	offline	CEN			
(Hoffmann et al., 2006)	IG	Local opt.	online	DIS	✓		
(Sarmiento et al., 2009)	**ET**	DFS/NR	offline	CEN			
(Lin and Goodrich, 2009)	P_d	GA/Greedy/LH	offline	CEN			
(Tisdale et al., 2009)	P_d	Gradient-based	online	DIS		✓	
(Gan and Sukkarieh, 2010)	$1-P_d$	Gradient-based	online	DIS			
(Delle Fave et al., 2010)	P_d	Max-sum	online	DIS			
(Lanillos et al., 2012)	**ET**/**DTR**	CEO	offline	CEN			
(Lo et al., 2012)	P_d	MIQP	offline	CEN			
(Lanillos et al., 2013)	**ET**/**DTR**	BOA	offline	CEN			
(Carpin et al., 2013)	$IG + distance$	Greedy	online	CEN			
(Lanillos et al., 2014a)	ET	CEO	offline	CEN			
(Lanillos et al., 2014b)	$\prod_{k=1}^{Q}(1-P_d^k)$	Gradient-based	online	DIS		✓	
(Berger et al., 2014)	$\sum_{k=1}^{Q} w_k P_d^k$	MIQP	offline	CEN			
(Khan et al., 2015)	dist. for $\bar{b}(v^t) > threshold$	NN/GA	online	both			
(Chang-jian et al., 2015)	$P_d + IG + Coop.$	MPC	online	DIS	✓		
(Berger et al., 2016)	P_d	MIQP	offline	CEN			
(Meghjani et al., 2016)	**MTTF**	Greedy/Spiral	offline	CEN			
(Raap et al., 2016)	P_d	BILP	online	CEN			
(Pérez-Carabaza et al., 2016b)	**MO** − **ET**	GA	offline	CEN	✓	✓	
(Yao et al., 2017)	P_d	GMM-RHC	offline	DIS	✓		
(Pérez-Carabaza et al., 2017)	**ET**	ACOR	offline	CEN	✓		✓
(San Juan et al., 2018)	weighted P_d	Fuzzy/PSO	offline	DIS			
(Pérez-Carabaza et al., 2018)	**ET**	MMAS	offline	CEN			✓
Chapter 4	**ET**	MMAS	offline	CEN			✓
Chapter 5	**MO** − **ET**	ACOR/GA	offline	CEN	✓	✓	✓

target along the UAV trajectories) is considered, the objective function is instead minimized. A different strategy related to P_d is followed by (Khan et al., 2015), which minimizes the distance to the cells whose probability of target presence is over a certain threshold. In the multi-target approaches where a unique belief describes the targets state (since they present an homogeneous behavior), the probability of finding as many targets as possible is maximized. However, when there are targets with heterogeneous characteristics, each target has a different belief or motion model, and hence the probability of detection P_d^k of each target k for a given set of UAV trajectories can differ from the probability of detection P_d^l of the others ($l \neq k$) for the same set of UAV trajectories. This implies that a set of UAV trajectories can be good for finding one target but not for exploring any area of interest of the other targets. Hence, in order to find an overall good set of UAV trajectories, (Wong et al., 2005) and (Berger et al., 2014) select as objective function a weighted sum of the probability of detection of each of the targets. Alternatively, (Lanillos et al., 2014b) minimizes the joint non-detection probability, that is, the probability of no detecting any of the targets, that can be computed, assuming target independence, as the product of the probability of non detecting a target along the trajectory.

- Another strategy is to minimize the a posteriori entropy of the target state distribution. The entropy of the belief can be interpreted as a measure of its uncertainty, therefore a high entropy value of the initial belief would correspond to a belief with a target presence probability quite spread over the search area, while a low entropy value indicates that most of the cells of the belief either have low or high changes of target presence (i.e. that the probability is concentrated). To optimize this objective (Hoffmann et al., 2006) maximize the information gain (IG), which is the difference between the previous entropy of the belief and the posteriori entropy after performing the sensor measurements. Hence, maximizing IG is equivalent to minimizing the posteriori entropy. It is worth noting that while optimization the information gain (also known as mutual information) implies that the UAVs distribute to cover the areas where we are

more uncertain about if the target is present or not, optimizing the probability of detection prioritizes the cells where we are more certain about the target presence. To accomplish both objectives, (Yang et al., 2002) optimize both through a linear combination, and (Chang-jian et al., 2015) adds also a third cooperation objective that reinforces the spread of the UAVs inside the search area. Besides, (Carpin et al., 2013), which uses a UAV waypoints dynamical model, computes the IG corresponding to all the cells of the map (possible next destinies of the UAV) and combines it with a function of the distance to each cell to penalize the time required to reach far away cells.

- None of the previously mentioned fitness functions ensure finding the targets in minimum time, and although may be appropriate objectives to other probabilistic search problems, in this thesis we focus on Minimum Time Search. Therefore, we have highlighted in bold in Table 2.3 the objective functions specific for MTS problems. Due to the uncertainty associated to the problem it is not possible to obtain the exact target detection time of a solution (set of UAV trajectories proposal), but its expected value can be calculated instead. For this reason, (Lanillos et al., 2014a; Pérez-Carabaza et al., 2017; Sarmiento et al., 2009) and (Pérez-Carabaza et al., 2018) optimize directly the expected value of the target detection time (in short the Expected Time, *ET*) . Besides, (Lanillos et al., 2012) and (Lanillos et al., 2013) consider two strategies: the ET and the Discounted Time Reward (*DTR*), obtaining this last one assigning decreasing weights to the probability gathered by the UAVs at each time step. DTR strategy is in fact a modification of the P_d strategy: while in P_d all the measurements have equal importance and their time order does not matter, DTR gives more importance to the early measurements with the purpose of indirectly minimizing the search time. Besides, (Lanillos et al., 2013) analyzes the ET of the solutions obtained with the ET and DTR strategy over several scenarios and concludes that the solutions obtained with ET are usually, at least, as good as those obtained with DTR. Alternatively, the strategy used in (San Juan et al., 2018) weights P_d by the visiting order of the cells, and (Pérez-Carabaza et al., 2016b) proposes a

multi-objective strategy (labelled as *MO-ET*) that considers the ET, a myopia heuristic criterion, the fuel consumption and the smoothness of the trajectories. Finally, (Meghjani et al., 2016) computes the mean of the finding times (in short the mean time to find, MTTF) obtained in several Monte Carlo simulations that start with different initial target positions sampled from the initial belief. Although this strategy may be adequate to evaluate the single solution generated in some deterministic planners, it is computationally expensive for evaluating the solutions of optimization approaches that create and analyze multiple solutions. Hence, in these cases, computing the ET of the solutions is a much faster approach than running several simulations to compute their MTTF.

The third column of Table 2.3 contains the wide number and types of *optimization methods* used to find good search trajectories. (Eagle, 1984) formulates the problem as a Partially Observable Markov Decision Process (*POMDP*) and solves it with a Dynamic Programming (DP) technique. This method is the only one that returns a global optimal solution, but it is only applicable to very simple scenarios (and in that paper it is tested over a 3x3 cells grid scenario with a unique UAV). Due to the high complexity of the PS problems, they are commonly solved with approximated methods such as Genetic Algorithms (*GA*, (Lin and Goodrich, 2009; Pérez-Carabaza et al., 2016b)), Bayesian Optimization Algorithm (*BOA*, Lanillos et al. (2013)), greedy methods ((Bourgault et al., 2006) and (Carpin et al., 2013)), local optimization ((Hoffmann et al., 2006)), Cross Entropy Optimization (CEO, (Lanillos et al., 2014a, 2012)), reinforcement learning techniques (Q-learning, (Yang et al., 2002)), Particle Swarm Optimization (PSO, (San Juan et al., 2018)) or gradient-based methods ((Gan and Sukkarieh, 2010; Tisdale et al., 2009) and Lanillos et al. (2014b)). Alternatively, (Wong et al., 2005) uses a constraint non linear programming technique called Sequential Quadratic Programming (*SQP*). (Sarmiento et al., 2009) divides the search problem inside a building in two levels; in the top level it decides the order of visiting the rooms as a combinatorial problem with the Depth-First Search (DFS) algorithm and in the low planning level it obtains the robot trajectory as a numerical problem using the Newton-Raphson (*NR*) method. (Delle Fave et al.,

2010) employs the max-sum algorithm as a distributed coordination technique to sequentially choose the actions of each UAV. (Khan et al., 2015) employs the Nearest Neighbour (*NN*) heuristic for solving the problem in single UAV scenarios and GA for solving the problems with multiple UAVs. (Yao et al., 2017) employs a Gaussian Mixture Model (*GMM*) to divide the belief in several subregions based on their probability that are then assigned to the UAVs based on their predicted payoff. Furthermore, (Lo et al., 2012) and (Berger et al., 2016) formulate the problem as a Mixed-Integer Linear and Quadratic Programming (*MIQP*), (Chang-jian et al., 2015) as a Model Predictive Controller (*MPC*) and (Raap et al., 2016) as a Binary Integer Linear Programming (*BILP*) and solve it based on max-k-coverage problem (Khuller et al., 1999). Alternatively, (Meghjani et al., 2016) and (San Juan et al., 2018) propose several deterministic heuristics: (Meghjani et al., 2016) use spiral patterns or methods that direct the UAV towards the cell with maximum belief, and the methods proposed by (San Juan et al., 2018) for single UAV problems direct the UAV towards the cell of the belief with maximum reward, calculated for each cell combining the belief and distance to the UAV with fuzzy-logic techniques. Finally, we propose a multi-objective algorithm based on GA in (Pérez-Carabaza et al., 2016b) and two algorithms based on ant colony techniques; Max-Min Ant System (*MMAS*) for the discrete UAV motion model (Pérez-Carabaza et al., 2018) and Ant Colony Optimization for Real-coded domains (*ACOR*) for the continuous UAV motion model (Pérez-Carabaza et al., 2017).

Furthermore, we can distinguish between offline and online planning in the *offline/online* column of Table 2.3. Receding horizon controller (RHC) is a widely used simplification assumed in both online and offline approaches, which divides the optimization of the whole trajectory in several subproblems, where each part of the trajectories is optimized consecutively and the UAVs final positions of each subproblem are used as the initial position of the following optimization subproblem. In this regard, all the analyzed online approaches make use of the RHC technique, performing the optimization during the execution of a previous planned search path. Hence, the maximum possible computational time is limited by the time required to

fly each of the sub-trajectories returned by the RHC. On the contrary, in the offline approaches the maximum computational time is less strict and therefore allows to find higher quality solutions. Even though, due to the high complexity of the problem, many of the offline works also applied RHC to reduce the search space and therefore the computational time (e.g (Wong et al., 2005) or (Pérez-Carabaza et al., 2016b)), while other works compute the full trajectory at once ((Eagle, 1984) or (Pérez-Carabaza et al., 2018)).

Another way of classifying the works, which is summarized in the CEN/DIS column of Table 2.3, is related to their computation distribution. In this regard, the UAV trajectories can be computed in a centralized way (*CEN*) in a base station or by the main UAV, or each UAV can compute its own plan in a distributed fashion (DIS) based on the available information. Descentralized or *distributed planning* has the advantage of not depending on a unique UAV or GCS, so it is more robust in case of failure of the main computing unit. On the other hand, *centralized planning* has a better ability to global decision-making, as it optimizes the paths of all the UAVs as a whole, exploiting all the available information simultaneously. Note that, good search paths computed independently by each UAV may not be good for global planning as the actions of one UAV may affect the utility of another (coupling problem). In other words, the information gathered by several UAVs when exploring a specific zone is less than the sum of the information gathered by each one independently. Nevertheless, (Wong et al., 2005) adopts a coordination simplified approach where each UAV optimizes its path without considering the trajectories of the others, while (San Juan et al., 2018) undertakes the cooperation problem by assigning to each UAV a portion of the search area. The rest of the distributed planning algorithms try to avoid the coupling problem through two different strategies: a) each UAV can try to predict the plans of the rest of the group and decide its own plan taking into account the predicted plans of the others, or b) the UAVs can send their plans to other UAVs and negotiate a common plan. For example, within the first group, (Yang et al., 2002) uses a neural network trained by reinforcement learning techniques to predict the actions of the rest of the UAVs, and within the second group, (Hoffmann et al., 2006)

uses an iterative negotiation process where in hierarchical order each UAV optimizes its path conditioned by the paths of the others (that are considered as fixed).

In multiple searcher scenarios, the problem of avoiding possible *collisions* between the searchers arises. This allows to distinguish the works also by the strategies that they use to ensure collision avoidance. Most of them avoid this problem simply by assuming that the UAVs operate at different heights, while others (indicated with a tick mark in the sixth column of Table 2.3) implement a method to explicitly check the possible collisions. In particular, (Hoffmann et al., 2006) and (Chang-jian et al., 2015) incorporate the collision constraint with a penalty method that adds its value to the objective function, while the planners by (Pérez-Carabaza et al., 2016b) and (Pérez-Carabaza et al., 2017) prefer feasible solutions over unfeasible ones (which are the UAVs trajectories that do not always keep a security distance between UAVs).

As previously stated, many works utilize the RHC technique, that is, they divide the optimization of the whole trajectory in several subproblems. This technique has the advantage of greatly reducing the solution space and therefore the computational time, but it may lead to myopic solutions. The key idea of myopia is that the best option in the current optimization horizon may not be a good option in the next one, that is, in RHC only the reward of the current optimization step is optimized, without considering if this option is beneficial in the following optimization steps. This effect is made worse when the lookahead depth is very short, which happens more often in online approaches due to the stricter limited computational times (for instance (Yang et al., 2002) uses only a horizon of 2 time steps and (Hoffmann et al., 2006) or (Carpin et al., 2013) of 1). In order to avoid myopia, some works include a *myopia heuristic* that helps to reduce the myopia effect caused by RHC (indicated with a tick mark in the seventh column of Table 2.3). We can divide the myopic heuristics into two groups: the ones that modify the optimization horizon or strategy in some specific myopic situations and the ones that estimate the future reward and consider this estimation in the evaluation of the solutions. The first group solves the myopic situation where there is any relevant probability area inside the horizon length of the

UAVs. For example, imagine that the UAV starts the search in an initial position that is further than the optimization horizon from the unique high probability area, in this situation all the solutions of the first horizon step would have similar fitness and the algorithm would not be able to distinguish that the best option is to get as closer as possible to the high probability area. (Tisdale et al., 2009) and (Wong et al., 2005) identified this myopic situation when the fitness value of the best solution is below a threshold, and to amend it, the former increases the optimization horizon and the later directs the UAV to the closer mode of the belief. The second group of heuristics utilizes some function to estimate long term rewards: (Yang et al., 2002) heuristic is based on an average of the rewards that are reachable by the UAVs in the future steps, (Lanillos et al., 2014b) heuristic is modelled as a long term sensor, and (Pérez-Carabaza et al., 2016b) weights the possible reachable belief with a function of the distance to the final positions of the UAVs at current optimized step.

The last category, represented in the last column of Table 2.3, indicates the works whose optimization methods consider problem specific heuristic information while constructing their solutions. Although the greedy strategies proposed in (Meghjani et al., 2016) and the fuzzy-logic based methods proposed in (San Juan et al., 2018) consider information about the scenario (distance from the UAV and belief probabilities), they are not categorized under the *constructive heuristics* category because they have not been embedded in an optimization method. On the contrary, the ant colony based optimization methods we propose in (Pérez-Carabaza et al., 2017) and (Pérez-Carabaza et al., 2018) benefit from the use of MTS constructive heuristics that guide the search during the optimization process and allow the algorithms to obtain high quality solutions in less computational time.

To sum up, we propose two MTS algorithmic solutions based on ant colony optimization techniques that minimize the expected target detection time (ET) and that take advantage of the capacity of these techniques to include problem specific heuristics. The use of the constructive heuristics should accelerate the algorithms convergence allowing to increase the optimization horizon and therefore, indirectly reducing the myopia of the solutions. Moreover, although in (Pérez-Carabaza et al., 2017)

and (Pérez-Carabaza et al., 2018) the algorithms are implemented in a single-step optimization planner (in order to analyze in a clearer way the heuristics effects), they can be also implemented within a RHC approach and in combination with a myopic heuristic, as the results of Chapter 5 show.

2.3. Summary

In this chapter we have put into context the MTS problem with other related problems and its historical background, which had its origins during World War II. Then, we have analyzed in more detail the closer and more relevant probabilistic search works that have motivated this thesis.

Probabilistic search is related to path planning problems, with several common methods and codifications of solutions (trajectories). However, PS is not goal oriented as usual path planning and handles probabilistic information about the target. Another important family of related problems is coverage. In both problems one or several unmanned vehicles explore an area of interest, bus as generally in PS the resources and time are limited to explore the whole area, PS focuses the search on the more relevant zones. Besides, the analysis of other related problems is interesting because the techniques applied to solve them can often be adapted to the problem at hand. This is the case of this thesis, which proposes to apply ant colony based methods, widely known for solving the Travelling Salesman Problem, to Minimum Time Search.

The search problem has been approached from different disciplines and it can be noticed in the variety of methods and formulations used to tackled it. Works differ on many characteristics such as the way the target location information is modelled or the objective function optimized. With the purpose of clarifying the analysis, we have sequentially revised the features of the works attending to the search elements (target, UAVs and the search environment) as well as the main characteristics of the existing algorithms or planners. In this thesis, we propose multi-UAV search algorithms for static or dynamic targets based on ant colony techniques, that unlike

the state of the art works, take advantage of MTS constructive specific heuristics that guide the search towards high quality solutions accelerating the convergence of the algorithms to overall good solutions (UAV trajectories).

Chapter 3

Problem Formulation and Optimization Approach

"Essentially, all models are wrong, but some are useful"

George Box

In this chapter we introduce the mathematical formulation of Minimum Time Search (MTS). First, we state the MTS optimization problem and explain how the uncertain information of the elements involved is probabilistically modelled. Then, we introduce how the information about the target location is updated and how the search trajectories are evaluated. Next, we describe, from a general point of view, how PS algorithms solve the search problem. And finally, we introduce the ant colony optimization techniques that are chosen in this thesis to solve the MTS problem.

3.1. Problem Statement

In the Minimum Time Search (MTS) problem a group of U UAVs carries out the search for a single target with an unknown location, modelled by random variable v inside the search area Ω. Although the exact location of the target is unknown, it is very likely that there is some information about its location, such as in which zones

is more or less probable to find the target. Besides, in case that the target is not static, information about its dynamics may be also available.

The objective of the MTS is to determine the best UAVs search trajectories that minimize the time of detection of the target. The UAVs initial positions $s^0_{1:U}$ and their dynamical models are considered as known. The deterministic dynamical model of the u-th UAV $s^{t+1}_u = f(s^t_u, c^t_u)$ allows to calculate the UAV position at time step $t+1$ from its previous state at time step t and control action c^t_u, where $c^t_u \in \mathbf{C_u}$ and $\mathbf{C_u}$ is the domain of admissible actions of the u-th UAV. The use of a dynamical model has two benefits. On the one hand, it allows to determine the UAV trajectories (or sequence of UAV states $s^{1:T}_{1:U}$) from their initial states $s^0_{1:U}$ and sequences of control actions $c^{1:T}_{1:U}$. On the other hand, it ensures that the optimized search trajectories are feasible from the maneuverability point of view of each UAV. Besides, the UAVs trajectories length is determined by the time of the search mission T, which is considered as a given input and may be determined for instance by the fuel capacity of the UAVs.

Due to the uncertainty associated to the problem, the exact time of the detection of the target can not be computed, but it is possible to optimize its expected value instead. Therefore, the MTS problem can be formulated as:

$$\begin{aligned} &\text{minimize} \quad ET(s^{0:T}_{1:U}) \\ &\text{subject to} \quad s^{t+1}_u = f(s^t_u, c^t_u) \quad u = 1, \ldots, U \quad c^t_u \in \mathbf{C_u} \end{aligned} \tag{3.1}$$

The expected value of the target detection time (ET) of the trajectories of the UAVs depends of the available information about UAV sensors and the target location and dynamics. Besides, more complex MTS formulations can be considered: including complementary optimization objectives (e.g. UAV fuel consumption) or adding additional constraints (e.g. avoiding UAV collisions).

Furthermore, the deterministic behavior of the UAV dynamical model allows to optimize either the trajectory of the UAVs (Equation 3.1) or the sequence of control signals to be applied during the searching task (Equation 3.2). In order to optimize the control signals, the mobility of the UAVs is discretized in time by allowing the

vehicles to make decisions at discrete time steps and applying piecewise constant control sequences during fixed time intervals ΔT.

$$\begin{aligned} &\text{minimize} \quad ET(c_{1:U}^{1:N}) \\ &\text{subject to} \quad c_u^t \in \mathbf{C_u} \quad u = 1,\ldots,U \end{aligned} \tag{3.2}$$

where N is the planning horizon of the UAVs, defined as the number of control actions optimized for each UAV during the search task, which is obtained with Equation 3.3 dividing the total time of the search T by the time interval between consecutive actions of the UAV Δ.

$$N = T/\Delta T \tag{3.3}$$

The optimization of control signals has the advantage of ensuring that the search trajectories correspond to the UAV dynamical model. For this reason, this approach is taken in great part of state of the art works, e.g. (Gan and Sukkarieh, 2010) or (Lanillos et al., 2012). In this regard, it is especially interesting the work presented in (Lin and Goodrich, 2009), where both approaches are implemented with a Genetic Algorithm (GA), in one version the algorithm directly optimizes the trajectories (sequence of cells) and in the other the control actions (sequence of cardinal directions). However, the approach that optimizes directly the trajectories does not obtain significant better results and presents several implementation difficulties (as cross over can only be made in the intersection cells of the trajectories, the resulting trajectories have different lengths than their parents and have to be increased or cut).

3.2. Uncertainty Modelling

The main elements of the minimum time search problem are shown in the example scenario of Figure 3.1, where two UAVs ($U = 2$), equipped with sensors, are looking for a lost hiker. MTS algorithms optimize the UAV trajectories using location and dynamic information of the target, of the sensors and of the dynamic restrictions of the

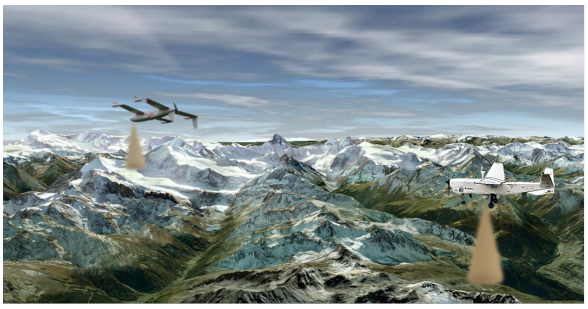


Figure 3.1 Search and rescue scenario where two UAVs are looking for a lost hiker.

UAVs. Due to its inherent uncertainty, the problem is tackled from a probabilistic approach and the information about the target and sensors is modelled probabilistically. However, as mentioned in the previous section, the UAV motion is modelled deterministically. The models of the different elements involved are introduced below: Section 3.2.1 presents the target models (target belief and target dynamical models) and Section 3.2.2 describes the UAV models (sensor likelihood and UAVs dynamical models).

3.2.1. Target Models

The target under search is initially located (at $t = 0$) in an unknown position $\mathbf{v}^0$ of the search area. The uncertain information about the target initial location is modelled with a belief or probability map $P(\mathbf{v}^0)$. Besides, the information about its dynamics is modelled with the target motion model $P(\mathbf{v}^t|\mathbf{v}^{t-1})$.

3.2.1.1. Probability Map or Belief

The information about the target location at any time step t is modelled with a probability map or belief $P(\mathbf{v}^t)$, which represents the probability of the target to be located in the different areas of the search region. The construction scheme of the initial probability map $P(\mathbf{v}^0)$, when $t = 0$, is shown in Figure 3.2. First, the search area is defined and discretized into a regular grid G_Ω of $(w_x \times w_y)$ cells, as shown in the two bottom layers of the figure. Then, a probability of target presence within a

cell is given for each of the cells in G_Ω based on available information (e.g. witnesses information, previous similar scenario, terrain elevation, etc). In the two top layers of Figure 3.2 the probability map is represented, where warmer colors indicate areas with higher chances of target presence, which should be explored as soon as possible in order to reduce the target detection time.

The probability map is a discretization of the target presence probability density function (PDF), where the value assigned to each cell corresponds to the integral of the PDF between the cell spatial limits. Furthermore, assuming that the target is inside the search area, the sum of the belief values of all the cells must sum up one initially (i.e. $\sum_{v^0 \in G_\Omega} P(v^0) = 1$) and during the whole mission (i.e. $\sum_{v^t \in G_\Omega} P(v^t) = 1$).

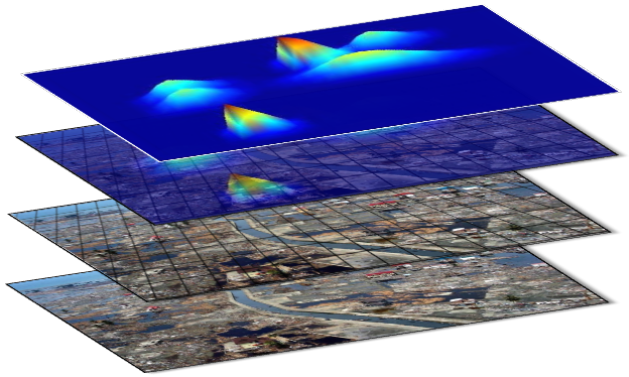


Figure 3.2 Probability map building process.

3.2.1.2. Target Dynamical Model

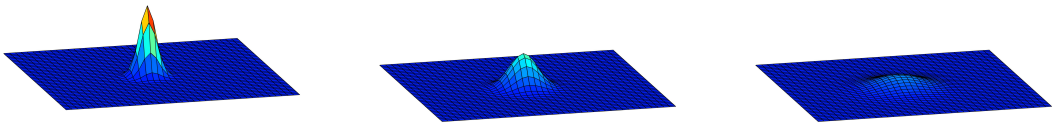
The information about the target dynamics is represented with the target motion probabilistic model $P(v^t|v^{t-1})$, which states the probability that the target moves from a cell v^t of the discretized region to its surrounding cells v^{t+1} at two consecutive time steps separated by ΔT interval.

When the target is static, the probability of staying in the same cell at two consecutive time steps is one, and the probability of moving to the surrounding cells is

zero:

$$P(\nu^t|\nu^{t-1}) = \begin{cases} 1 & \forall \nu^{t+1} = \nu^t \\ 0 & \forall \nu^{t+1} \neq \nu^t \end{cases} \tag{3.4}$$

When the target is dynamic and there is no information about its motion, a uniform spreading probability model can be used. An example of the effect of applying a uniform spreading model to a belief is shown in Figure 3.3, where the target initial belief is a centered gaussian and the dynamical model gives equal changes of either remaining in the same cell or moving to any of its neighbor cells. Hence, as time passes the probability that was initially concentrated in the center is spread over the search area. This simple example shows an interesting fact about PS with static versus moving targets. If the target is static there would not be any gain in revisiting areas that have been already efficiently explored, but when the target is dynamic, as time passes, it is possible that the target belief moves to an area that it was already efficiently explored, and therefore the UAVs may have to overfly it again.



(a) $P(\nu^0)$ (b) $P(\nu^5)$ (c) $P(\nu^{20})$

Figure 3.3 Target mono-Gaussian initial belief and later beliefs, at two different time instants, after applying a uniform spreading dynamical model.

The target motion model is assumed to be Markovian $P(\nu^t|\nu^{t-1})$, where the target movements only depend on their previous state. A common way to describe $P(\nu^t|\nu^{t-1})$ is with a transition matrix A of size $(w_x w_y \times w_x w_y)$, where the value of the element at i-th row and j-th column $A(i,j)$ indicates the probability of moving from cell j to the cell i (Eagle, 1984). Hence, in order to maintain constant the probability of target presence inside the search area the sum of the probabilities of all the possible movements from cell j to other cells (including itself) should be one

$$A = \begin{bmatrix} 1 & 0.4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.6 & 0.4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0.4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.6 & 0.4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.6 & 0.4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.6 \end{bmatrix}$$

(a) $P(v^t|v^{t-1})$

1	4	7
2	5	8
3	6	9

(b) Cells indexing

Figure 3.4 Target transition matrix for a target dynamical model with a probability of 0.4 of moving north and 0.6 of staying in the same cell.

(i.e. $\sum_{i=1,\ldots,w_x w_y} A(i,j) = 1 \ \ \forall j = 1,\ldots,w_x w_y$). As an illustrative example, the transition matrix displayed in Figure 3.4 (a) corresponds to a simple grid of $w_x = w_y = 3$ cells, and a target movement model with a probability of 0.6 of staying in the same cell and a probability of 0.4 of moving towards the north (if allowed). In the example, we consider column-wise numeration of the cells of the belief displayed in Figure 3.4 (b). Therefore, as for the first cell (first column) going north is not allowed, the probability of staying at the same cell $A(1,1) = 1$, for the second cell (second column) there is a probability of 0.6 of staying in the same cell (i.e $A(2,2) = 0.6$) and of 0.4 of moving north to cell 1 (i.e. $A(1,2) = 0.4$), etc.

It is worth noting that since only the movements from the current cell of the target to its neighbor cells are allowed, transition matrixes A are sparse, with relevant information saved only in a few of their elements. We take advantage of this propriety in order to have more efficient calculations and use less physical memory by only saving and considering in the calculations where $P(v^t|v^{t-1})$ is involved, the elements of $A(i,j)$ where the cells i and j are neighbor cells.

3.2.2. UAV Models

The search is carry out by UAVs with sensors capable of making detection measurements of the targets. First, we introduce the deterministic UAV dynamical models, which ensure that the trajectories are adequate for the UAVs dynamics. Then, we present the probabilistic sensor models, which allow to update the target belief with the sensor measurements accordingly to the sensor performance.

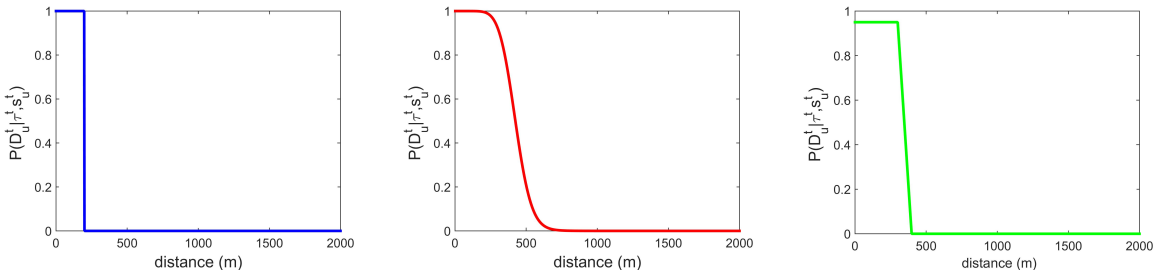
3.2.2.1. UAV Dynamical Model

The UAV motion model $s_u^{t+1} = f(s_u^t, c_u^t)$ lets us obtain the next position of a UAV s_u^{t+1} given the previous position s_u^t and the control action c_u^t at time t. The deterministic behaviour of the model allows to obtain the UAV trajectories either by the UAVs states $s_{1:U}^{1:N}$ or by their initial states $s_{1:U}^0$ and control actions $c_{1:U}^{1:N}$.

In this thesis we use two different UAV dynamic models:

- *Discrete cardinal model.* This model allows to move the UAV from its current position (centered in a cell of the search space at a given height) to the center of one of the neighbor cells following the actions defined by the cardinal directions $\mathbf{C_u} = \{N, NE, E, SE, S, SW, W, NW\}$. Therefore, the corresponding trajectories are sequences of centers of adjacent cells at fixed height, which can be followed by rotatory-wing UAVs.
- *Continuous Simulink model.* This is a differential non-linear kinematic model that produces smooth trajectories more appropriate for fixed-wing UAVs. The model can be adapted to the different characteristics of the UAVs (like maximum height or bank angle) and has a continuous action domain $\mathbf{C_u} \in \mathbb{R}$.

The different domain of the actions in each model (eight possible values in the discrete cardinal model and continuous domain in the continuous Simulink model) are better tackled by different families of optimization algorithms. For this reason, the discrete dynamical model is used in the MTS algorithms presented in Chapter 4 and



(a) Ideal (b) Radar (c) Camera

Figure 3.5 Several sensor models probability curves: evolution of the probability of target detection with the distance from the sensor to the target.

the continuous model in the approaches of Chapter 5. Further information of the models can be found in the respective chapters.

3.2.2.2. Sensor Models

The sensor model enables to represent the uncertainty associated with the sensor measurements. Each UAV is equipped with a sensor that takes measurements every ΔT and has associated a sensor model $P(z_u^t|v^t,s_u^t)$ that states the probability that the u-th UAV makes a certain measurement z_u^t conditioned by the sensor location and the target position v^t. Besides, as the deviation of the sensor location from the UAV location is negligible for the mission, the sensor location equals s_u^t.

A general assumption in PS works is considering two types of sensor measurements: target detection $z_u^t = D$ and non target detection $z_u^t = \overline{D}$. In this case, the probability of detecting the target $P(z_u^t = D|v^t,s_u^t) \equiv P(D_u^t|v^t,s_u^t)$ is complementary to the probability of non-detection, $P(z_u^t = \overline{D}|v^t,s_u^t) \equiv P(\overline{D}_u^t|v^t,s_u^t) = 1 - P(D_u^t|v^t,s_u^t)$.

In this thesis we utilize three different types of sensor models, whose sensor curves are displayed in Figure 3.5. The sensor curves show the evolution of the probability of target detection $P(D_u^t|v^t,s_u^t)$ with the distance from the sensor to the target:

- *Ideal sensor model.* An ideal sensor model that considers a probability of detection of 1 if the target is in the cell underneath the UAV's sensor and 0 otherwise.

This type of model is often used in PS works, for example in (Meghjani et al., 2016) or (San Juan et al., 2018). Figure 3.5 (a) shows the sensor curve of an ideal sensor model, which has a probability of detection of 1 when the target is underneath the UAV in a cell of 200×200 meters.

- *Radar sensor model.* The probability of detection of a radar decreases exponentially with the distance from the target to the sensor, and the maximum probability value can be obtained considering different characteristics of the radar and the target. Radar models are used in several works of the PS literature, e.g. in (Furukawa et al., 2003) or (Wong et al., 2005). The probability curve of the radar model of Figure3.5 (b) shows how the detection probability decreases from a maximum probability value of 1 as the distance from the sensor to the target increases.

- *Camera sensor model.* Camera sensor models have a certain probability of target detection if the target is inside the camera footprint and zero otherwise. Besides, as the probability curve of the camera model of Figure 3.5 (c) shows, some intermediate detection probability values may be considered for the cells that are only partially inside the footprint. Camera models are commonly used in PS literature, for instance in (Tisdale et al., 2009) or (Delle Fave et al., 2010).

In this thesis, the ideal sensor model is used to analyze the results of the discrete algorithms presented in Chapter 4, the radar model is employed to characterize the MTS algorithms with continuous UAV models presented in Chapter 5, and the camera model is used during the integration of the proposed MTS planner in Airbus simulator described in Chapter 6. Nevertheless, it is worth noting that all sensor models could be indistinctly used in all the algorithms presented in this thesis. Further details of the employed sensor models and the justification of their choice are given in the corresponding chapters.

3.3. Recursive Bayesian Filter (RBF)

The Recursive Bayesian Filter (RBF, (Furukawa et al., 2003)) is a recursive algorithm that enables to predict the belief state through the target dynamical model $P(v^t|v^{t-1})$ and update the target belief with new information of the sensor measurements through the sensor model $P(D_u^t|v^t, s_u^t)$.

The recursive estimation approach can be divided in two steps: prediction and update. In the prediction step, the belief $b(v^{t-1})$, defined by Equation 3.5 and updated with previous target movements and sensor measurements up to time step $t-1$, is updated to the next target position v^t obtaining $\hat{b}(v^t)$, which is defined by Equation 3.6. Then, in the update step, $\hat{b}(v^t)$ is updated with the new sensor measurements $z_{1:U}^t$ and UAV states $s_{1:U}^t$ obtaining $b(v^t)$, which is defined by Equation 3.7. Note that, at the beginning of the recursive process, when only the first time step measurements $z_{1:U}^0$ have been taken, $b(v^0) \triangleq P(v^0|z_{1:U}^0, s_{1:U}^0) \propto \prod_{u=1}^{U} P(z_u^0|v^0, s_u^0)P(v^0)$, where $P(v^0)$ is the initial probability map.

$$b(v^{t-1}) \triangleq P(v^{t-1}|z_{1:U}^{0:t-1}, s_{1:U}^{0:t-1}) \tag{3.5}$$

$$\hat{b}(v^t) \triangleq P(v^t|z_{1:U}^{0:t-1}, s_{1:U}^{0:t-1}) \tag{3.6}$$

$$b(v^t) \triangleq P(v^t|z_{1:U}^{0:t}, s_{1:U}^{0:t}) \tag{3.7}$$

Finally, note that RBF is used in most of the PS state of the art works, e.g. (Wong et al., 2005) or (Delle Fave et al., 2010). Although there are other filters that enable to update the state of a probabilistic variable, like the well known Kalman Filter (Kalman, 1960), RBF is especially suitable for PS formulations as it allows to maintain highly non-Gaussian general Probability Density Functions (PDFs) of the targets state considering non-linear process models and heterogeneous non-Gaussian sensor models (Tisdale et al., 2009).

Prediction Step

The prediction step is necessary in Bayesian analysis when the PDF of the target state evolves with the time. It updates the target belief from the previous time step considering the target dynamic information modelled by $P(v^t|v^{t-1})$.

$$\hat{b}(v^t) = \sum_{v^{t-1} \in G_\Omega} P(v^t|v^{t-1}) b(v^{t-1}) \tag{3.8}$$

Proof. The prediction step equation is obtained applying Chapman-Kolmogorov equation (Pillai and Papoulis, 2002) and assuming a Markov target motion model (i.e. that the target state only depends on its previous state $P(v^t|v^{t-1}, z_{1:U}^{0:t-1}, s_{1:U}^{0:t-1}) = P(v^t|v^{t-1})$).

$$\begin{aligned}
\hat{b}(v^t) = P(v^t|z_{1:U}^{0:t-1}, s_{1:U}^{0:t-1}) &= \sum_{v^{t-1} \in G_\Omega} P(v^t, v^{t-1}|z_{1:U}^{0:t-1}, s_{1:U}^{0:t-1}) = \\
&= \sum_{v^{t-1} \in G_\Omega} P(v^t|v^{t-1}, z_{1:U}^{0:t-1}, s_{1:U}^{0:t-1}) P(v^{t-1}|z_{1:U}^{0:t-1}, s_{1:U}^{0:t-1}) = \\
&= \sum_{v^{t-1} \in G_\Omega} P(v^t|v^{t-1}) P(v^{t-1}|z_{1:U}^{0:t-1}, s_{1:U}^{0:t-1}) = \\
&= \sum_{v^{t-1} \in G_\Omega} P(v^t|v^{t-1}) b(v^{t-1})
\end{aligned}$$

□

Update Step

Considering the sensor models, this step updates the belief obtained after the prediction step with new sensor measurements.

$$b(v^t) = \frac{1}{\xi} \prod_{u=1:U} P(z_u^t|v^t, s_u^t) \hat{b}(v^t) \tag{3.9}$$

where ξ is a normalization factor that ensures that the probability of target presence inside the search area is maintained constant (i.e. $\sum_{v^t \in G_\Omega} b(v^t) = 1$).

Proof. The following expression shows how Bayes Rule allows to express the a posteriori belief $b(\nu^t) = P(\nu^t|z_{1:U}^{0:t}, s_{1:U}^{0:t})$ in terms of the belief returned by the prediction step $\hat{b}(\nu^t) = P(\nu^t|z_{1:U}^{0:t-1}, s_{1:U}^{0:t})$. Besides, considering $P(z_{1:U}^t|z_{1:U}^{0:t-1}, s_{1:U}^{0:t})$ equal to the normalization constant ξ and assuming independence among sensor measurements we obtain the update state equation:

$$
\begin{aligned}
b(\nu^t) = P(\nu^t|z_{1:U}^{0:t}, s_{1:U}^{0:t}) &= \frac{P(z_{1:U}^t|z_{1:U}^{0:t-1}, \nu^t, s_{1:U}^{0:t}) P(\nu^t|z_{1:U}^{0:t-1}, s_{1:U}^{0:t})}{P(z_{1:U}^t|z_{1:U}^{0:t-1}, s_{1:U}^{0:t})} = \\
&= \frac{1}{\xi} P(z_{1:U}^t|\nu^t, s_{1:U}^t)\hat{b}(\nu^t) = \\
&= \frac{1}{\xi} \prod_{u=1:U} P(z_u^t|\nu^t, s_u^t)\hat{b}(\nu^t)
\end{aligned}
$$

□

Finally, it is worth mentioning that the initial value $b(\nu^0)$ is obtained as a special case of the update Equation 3.9 in order to incorporate the initial measurements $z_{1:U}^0$ to the initial probability map $P(\nu^0)$.

RBF Algorithm

The pseudocode of RBF is shown in Algorithm 1. The algorithm requires as inputs the initial belief or probability map $P(\nu^0)$, the target dynamic model $P(\nu^t|\nu^{t-1})$, the sensor models $P(z_u^t|\nu^t, s_u^t)$, the UAVs trajectories $s_{1:U}^{0:T}$, the sensor measurements $z_{1:U}^{0:T}$, the total time of the search mission T and the time interval between sensor measurements and between target movements ΔT (which are assumed to be the same). First, the belief is initialized with the initial probability map and the initial measurements (in line 1) and the total number of steps is computed dividing the total search time between the time interval (in line 2). Then, after recursively predicting the target movements and updating the sensor measurements for each time step (loop from line 3 to 6), RBF returns as output the updated belief $b(\nu^N)$ (in line 7).

Algorithm 1 RBF

Require: $P(v^0), P(v^t|v^{t-1}), P(z_u^t|v^t, s_u^t)$ ▷ Initial belief, target dynamic and sensor models
Require: $s_{1:U}^{0:T}$, $z_{1:U}^{0:T}$, T, ΔT ▷ Sensor measurements, UAVs states, search time, time interval

1: $b(v^0) = \frac{1}{\xi} \prod_{u=1:U} P(z_u^t|v^t, s_u^t) P(v^0)$ ▷ Initialize target belief
2: $N = T/\Delta T$ ▷ Number of time steps
3: **for** t=1:N **do**
4: $\quad \hat{b}(v^t) = = \sum_{v^{t-1} \in G_\Omega} P(v^t|v^{t-1}) b(v^{t-1})$ ▷ Prediction Step
5: $\quad b(v^t) = \frac{1}{\xi} \prod_{u=1:U} P(z_u^t|v^t, s_u^t) \hat{b}(v^t)$ ▷ Update Step
6: **end for**
7: **return** $b(v^N)$ ▷ Updated target belief up to $t = N$

Furthermore, the prediction and update steps (Equation 3.8 and Equation 3.9) can be combined in a unique expression (Equation 3.10) that obtains the belief $b(v^t)$ updating $b(v^{t-1})$ with the new possible measurements through the sensor model $P(z_u^t|v^t, s_u^t)$ and target movements defined by the target dynamical model $P(v^t|v^{t-1})$.

$$b(v^t) = P(v^t|z_{1:U}^{0:t}, s_{1:U}^{0:t}) = \frac{1}{\xi} \prod_{u=1:U} P(z_u^t|v^t, s_u^t) \sum_{v^{t-1} \in G_\Omega} P(v^t|v^{t-1}) b(v^{t-1}) \quad (3.10)$$

3.4. Evaluation of Search Trajectories

In this section we define several PS evaluation strategies and derive their expressions, which enable to obtain the fitness values using the input information of the search problem: initial UAVs states, target initial belief, target motion model and sensor models. First, the probability of the overall target detection from a given set of UAVs trajectories $s_{1:U}^{0:N}$ and its complementary probability (the joint probability of non target detection) are presented. Then, the expected value of target detection time (ET) is presented and its expression (that depends on the joint probability of non-detection) is derived.

3.4.1. Maximizing the Probability of Target Detection

Maximizing the probability of target detection is the most commonly used strategy in PS state of the art, e.g. (Tisdale et al., 2009) or (Yao et al., 2017). The probability of detection $P_d(s_{1:U}^{0:N})$ when the UAVs follow their trajectories $s_{1:U}^{0:N}$ is defined as the union of the probability of target detection measurements $z_u^t = D$ along $s_{1:U}^{0:N}$. In other words, the probability of target detection $P_d(s_{1:U}^{0:N})$, defined by Equation 3.11, is the probability that the target is detected at some point along any of the UAV trajectories.

$$P_d(s_{1:U}^{0:N}) \triangleq P(\bigcup_{t=0:N,u=1:U} D_u^t | s_{1:U}^{0:N}) \tag{3.11}$$

Maximizing the probability of target detection is equivalent to minimizing the joint probability of failing to detect the target in all the time instants from the UAVs trajectories, i.e. the joint probability (intersection) of non detection measurements $P_{nd}(s_{1:U}^{0:N})$, defined by Equation 3.12.

$$P_{nd}(s_{1:U}^{0:N}) \triangleq P(\bigcap_{t=0:N,u=1:U} \overline{D}_u^t | s_{1:U}^{0:N}) = P(\overline{D}_{1:U}^{0:N} | s_{1:U}^{0:N}) \tag{3.12}$$

Proof. The relationship between both probabilities can be proven by applying the complementary operator and De Morgan's law.

$$\begin{aligned} P_d(s_{1:U}^{0:N}) \triangleq & P(\bigcup_{t=0:N,u=1:U} D_u^t | s_{1:U}^{0:N}) = 1 - P(\overline{\bigcup_{t=0:N,u=1:U} D_u^t} | s_{1:U}^{0:N}) = \\ = & 1 - P(\bigcap_{t=0:N,u=1:U} \overline{D}_u^t | s_{1:U}^{0:N}) = 1 - P_{nd}(s_{1:U}^{0:N}) \end{aligned} \tag{3.13}$$

□

Furthermore, the joint probability of non-detection $P_{nd}(s_{1:U}^{0:N})$ can be obtained with Equations 3.14, 3.15 and 3.16.

$$P_{nd}(s_{1:U}^{0:N}) = \sum_{v^N \in G_\Omega} \tilde{b}(v^N) \tag{3.14}$$

$$\tilde{b}(\nu^t) \triangleq P(\overline{D}_{1:U}^{0:t}, \nu^t | s_{1:U}^{0:t}) = \prod_{u=1:U} P(\overline{D}_u^t | \nu^t, s_u^t) \sum_{\nu^{t-1} \in G_\Omega} P(\nu^t | \nu^{t-1}) \tilde{b}(\nu^{t-1}) \quad (3.15)$$

$$\tilde{b}(\nu^0) = \prod_{u=1:U} P(\overline{D}_u^0 | \nu^0, s_u^0) P(\nu^0) \quad (3.16)$$

The joint probability of non-target detection $P_{nd}(s_{1:U}^{0:N})$ is obtained summing up the values of $\tilde{b}(\nu^t)$ for all the cells in G_Ω, where $\tilde{b}(\nu^t)$ is obtained recursively with Equation 3.15, using for the initial case $\tilde{b}(\nu^0)$ Equation 3.16, where $P(\nu^0)$ is the initial probability map. In fact, the recursive Equation 3.15, as it can be seen from its comparison with Equation 3.10, is a RBF for non detection measurements $z_{1:U}^{0:N} = \overline{D}_{1:U}^{0:N}$ without the normalization factor ξ. Moreover, the relationship of $\tilde{b}(\nu^t)$ with $b(\nu^t)$ makes us call the former the "unnormalized belief", understanding that $\tilde{b}(\nu^t)$ does not constitute a real belief as $\sum_{\nu^t \in G_\Omega} \tilde{b}(\nu^t) \neq 1$ due to the lack of the normalization term in Equation 3.15.

Proof. In order to obtain Equation 3.14, we marginalize over ν^N.

$$P_{nd}(s_{1:U}^{0:N}) = P(\overline{D}_{1:U}^{0:N} | s_{1:U}^{0:N}) = \sum_{\nu^N \in G_\Omega} P(\overline{D}_{1:U}^{0:N}, \nu^N | s_{1:U}^{0:N}) = \sum_{\nu^N \in G_\Omega} \tilde{b}(\nu^N)$$

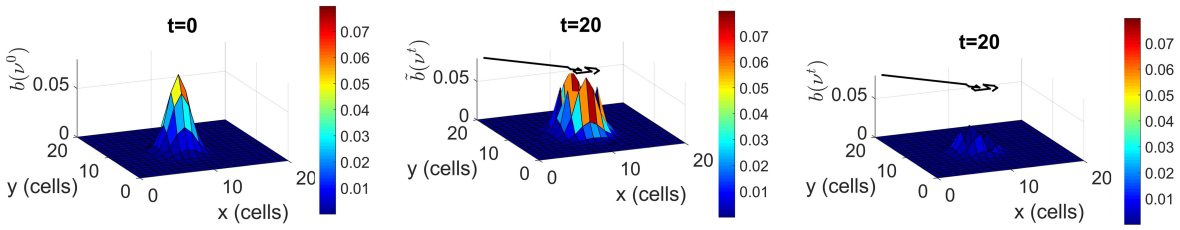
To obtain recursive Equation 3.15 that expresses $\tilde{b}(\nu^t)$ in terms of the problem probability models and previous time step "unnormalized belief" $\tilde{b}(\nu^{t-1})$, first the conditional probability operator is applied, second the conditional independence of each measurement on everything except ν^t and s_u^t is assumed, third the marginalization operation over ν^{t-1} is used, forth the conditional probability operator applied,

and finally a Markovian target motion is assumed.

$$\begin{aligned}
\tilde{b}(v^t) =& P(\overline{D}_{1:U}^{0:t}, v^t | s_{1:U}^{0:t}) = P(\overline{D}_{1:U}^{t} | \overline{D}_{1:U}^{0:t-1}, v^t, s_{1:U}^{0:t}) P(\overline{D}_{1:U}^{0:t-1}, v^t | s_{1:U}^{0:t}) = \\
&= \prod_{u=1:U} P(\overline{D}_u^t | v^t, s_u^t) P(\overline{D}_{1:U}^{0:t-1}, v^t | s_{1:U}^{0:t}) = \\
&= \prod_{u=1:U} P(\overline{D}_u^t | v^t, s_u^t) \sum_{v^{t-1} \in G_\Omega} P(\overline{D}_{1:U}^{0:t-1}, v^t, v^{t-1} | s_{1:U}^{0:t}) = \\
&= \prod_{u=1:U} P(\overline{D}_u^t | v^t, s_u^t) \sum_{v^{t-1} \in G_\Omega} P(v^t | \overline{D}_{1:U}^{0:t-1}, v^{t-1}, s_{1:U}^{0:t}) P(\overline{D}_{1:U}^{0:t-1}, v^{t-1} | s_{1:U}^{0:t-1}) = \\
&= \prod_{u=1:U} P(\overline{D}_u^t | v^t, s_u^t) \sum_{v^{t-1} \in G_\Omega} P(v^t | v^{t-1}) \tilde{b}(v^{t-1})
\end{aligned}$$

Finally, Equation 3.16 for the initial $\tilde{b}(v^0)$ is a special case of Equation 3.15, where only the starting non-detection measurements have to be incorporated to the initial probability map $P(v^0)$. □

Figure 3.6 displays, from left to right, a gaussian initial probability $P(v^0)$ of a static target and the corresponding $b(v^t)$ and $\tilde{b}(v^t)$ obtained with an ideal sensor from the search trajectory of a unique UAV represented by the black line. While the belief displayed in Figure 3.6 (b) is obtained with a RBF assuming non-detection measurements, the "unnormalized belief" of Figure 3.6 (c) is obtained with Equation 3.15, which is a RBF without normalization step with the assumption of non-detection mea-

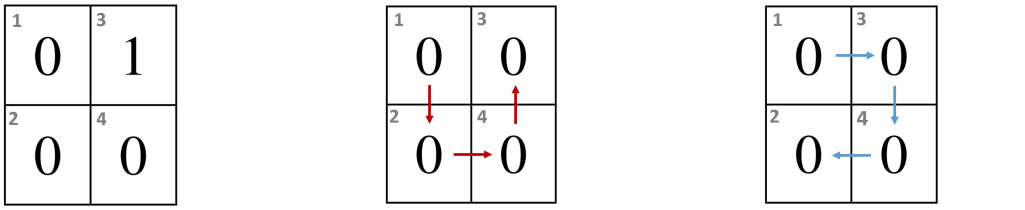


(a) $b(v^0)$ (b) $b(v^t)$ (c) $\tilde{b}(v^t)$

Figure 3.6 From left to right, initial belief $b(v^0)$ and updated belief $b(v^{20})$ and "unnormalized belief" $\tilde{b}(v^{20})$ with a UAV search trajectory (black line).

surements. Therefore, while the belief $b(\nu^t)$ is a probability (i.e. $\sum_{\nu^t \in G_\Omega} b(\nu^t) = 1$), the "unnormalized belief" is not. Besides, according to Equation 3.14, $P_{nd}(s_{1:U}^{0:t}) = \sum_{\nu^t \in G_\Omega} \tilde{b}(\nu^t) = 0.55 < 1$, and the probability of detecting the target corresponding to the trajectory displayed with a black line is, according to Equation 3.13, $P_d(s_{1:U}^{0:t}) = 1 - P_{nd}(s_{1:U}^{0:t}) = 0.65$.

Although maximizing the probability of detection $P_d(s_{1:U}^{0:t})$ is a good strategy when we are interested in maximizing the chances of finding the target for a given time horizon, it is not the best strategy for MTS problems, as it does not ensure the minimization of the target detection time. This can be observed in the example of Figure 3.7, which represents two possible search trajectories (the red in the center graphic and the blue in the right one) for a unique UAV equipped with an ideal sensor that starts the search at $s_1^0 = 1$ of a search region of only four cells (numbered column wise), with all the probability initially concentrated in the third cell (according to the graphic on the left). For the given planning horizon of $N = 3$, both trajectories have equal chances of finding the target $P_d(s_{1:U}^{0:N}) = 1$, but as we can intuitively expect, a UAV that follows the blue trajectory, which overflies the third cell in the first time step, has more chances of finding the target sooner than a UAV that follows the red trajectory, which overflies the third cell in the final time step. In fact, for this simple scenario, where all the probability is concentrated in one cell, we expect to find the target at the third time step when following the trajectory of Figure 3.7 (b) and in the first time step with the one of Figure 3.7 (c). Therefore, in this thesis, we do not optimize $P_d(s_{1:U}^{0:N})$ but instead, the expected target detection time ET. However, we use $P_d(s_{1:U}^{0:t})$ and the resulting "unnormalized belief" $\tilde{b}(\nu^t)$ to illustrate the results of the proposed algorithms as they allow to quickly understand the quality of a search trajectory: either by the value of $P_d(s_{1:U}^{0:t})$ (which quantifies the gathered information) or by the visual comparison of $\tilde{b}(\nu^t)$ with the initial belief $b(\nu^0)$ (which allows to observe visually the explored areas).



(a) $b(v^0)$ (b) $\tilde{b}(v^N)$, $s_1^{0:N} = \{1,2,4,3\}$ (c) $\tilde{b}(v^N)$, $s_1^{0:N} = \{1,3,4,2\}$

Figure 3.7 Example with two search trajectories in a simple search scenario where one UAV carries out the search, the considered planning horizon is N=3 and $w_x = w_y = 3$.

3.4.2. Minimizing the Searching Time

The main objective of the MTS problem is to determine the UAV search trajectories that minimize the searching time, i.e. the trajectories that allow to detect the target as soon as possible. Due to the uncertainty associated to the problem (sensor behavior and target location and dynamics), the exact detection time of a given search trajectory can not be determined. Instead, its expected value can be used.

3.4.2.1. Expected Target Detection Time

The Expected target detection Time (ET) given by Equation 3.17 is a common strategy optimized in MTS problems, e.g. (Sarmiento et al., 2009) or (Lanillos et al., 2012).

$$ET(s_{1:U}^{0:N}) = \sum_{t=0}^{N} P_{nd}(s_{1:U}^{0:t}) \tag{3.17}$$

Proof. Defining the time of detection of the target as a positive continuous random variable T, its expected value can be computed as

$$E\{T\} = \int_0^{\infty} (1 - P(T \leq t))dt \tag{3.18}$$

where $P(T \leq t)$ is the probability distribution function of the random variable T.

Besides, assuming that sensor measurements take place at discrete time instants equidistant by one second ($\Delta T = 1$), the expected time of finding the target is given

by Equation 3.19.

$$E\{T\} = \sum_{t=0}^{\infty}(1 - P(T \leq t)) \tag{3.19}$$

However, the computation of infinite terms is intractable (Bourgault et al., 2006) due to the limited computing capacity and to the limited resources available for the search (e.g. the fuel capacity of the UAVs). Therefore, a truncated version of Equation 3.18 is usually optimized.

$$E\{T\} = \sum_{t=0}^{N}(1 - P(T \leq t)) \tag{3.20}$$

Although, this truncated version returns a lower estimation of the expected value of the target detection, it is in fact an optimal policy[1] for the decision horizon N (Lanillos, 2013). For this reason, here-after we would indifferently refer to ET or its truncated version.

The expected time of target detection is computed adding up for each time step the probability $P(T \leq t)$ of detecting the target up to time instant t. This probability has been already defined as $P_d(s_{1:U}^{0:t}) = P(\bigcup_{l=0:t,u=1:U} D_u^l | s_{1:U}^{0:t})$ in Equation 3.11.

$$ET(s_{1:U}^{0:N}) = \sum_{t=0}^{N}(1 - P_d(s_{1:U}^{0:t})) \tag{3.21}$$

Moreover, we can substitute $P_d(s_{1:U}^{0:t})$ by its complementary $P_{nd}(s_{1:U}^{0:t})$, the probability of having all non-detection measurement along the search trajectory, Equation 3.13.

$$ET(s_{1:U}^{0:N}) = \sum_{t=0}^{N}(1 - (1 - P_{nd}(s_{1:U}^{0:t})) = \sum_{t=0}^{N} P_{nd}(s_{1:U}^{0:t}) \tag{3.22}$$

[1]Moreover, the truncated expression $E\{T\} = \sum_{t=0}^{N} t \cdot P(T = t)$ that computes the expected value in term of the density function instead of the distribution function (Bourgault et al., 2006) is not an optimal policy for the decision horizon N (Feller, 1968).

Finally, to change the measurement time step from 1 second to a different value, we only have to scale the $ET(s_{1:U}^{0:N})$ obtained by Equation 3.22 by the new measurement time lag ΔT. □

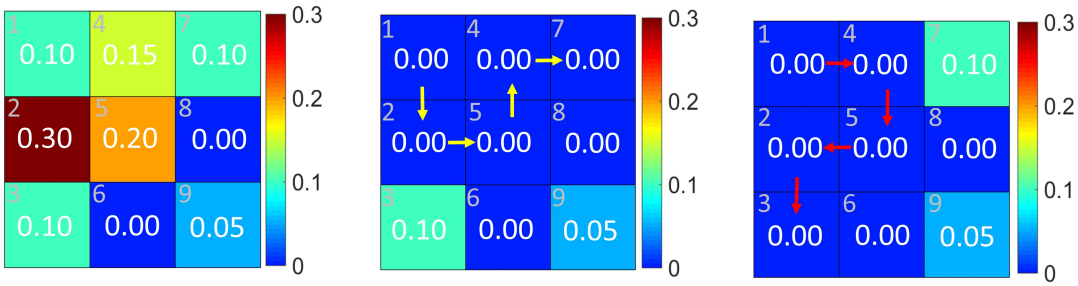
Using the Equation 3.15 previously derived to calculate $P_{nd}(s_{1:U}^{0:t})$, we can express the expected detection time in terms of the input information of the MTS problem. The expected time of a given set of UAV search trajectories $ET(s_{1:U}^{1:N})$ can be obtained with the recursive Equation 3.23, adding up for each time step up to the horizon N the "unnormalized belief" updated with the target dynamical information and sensor measurements, and initially valued as $\tilde{b}(\nu^0) = \prod_{u=1:U} P(\overline{D}_u^0|\nu^0, s_u^0) b(\nu^0)$.

$$ET(s_{1:U}^{0:N}) = \sum_{t=0}^{N} \sum_{\nu^t \in G_\Omega} \tilde{b}(\nu^t) = \sum_{t=0}^{N} \sum_{\nu^t \in G_\Omega} \prod_{u=1}^{U} P(\overline{D}_u^t|\nu^t, s_u^t) \sum_{\nu^{t-1} \in G_\Omega} P(\nu^t|\nu^{t-1}) \tilde{b}(\nu^{t-1}) \tag{3.23}$$

Finally, although the MTS main objective is to minimize the target detection time, the optimization of the expected time can be complemented with other objectives. For instance, in (Pérez-Carabaza et al., 2016b) the expected time is optimized in conjunction with other objectives such as fuel consumption or a myopia correction criterion.

3.4.2.2. Illustrative Example

To illustrate and evaluate our approach we will consider a simple scenario where the initial location density function of a static target is modelled by the initial probability map shown in Figure 3.8 (a). Besides, a unique UAV carries out the search starting from the upper left corner cell ($s_1^0 = 1$, cells are numbered columnwise) and with a planning horizon limited to $N = 4$ actions. The UAV is allowed to move from its current cell to the neighbor cells and its sensor performance is modelled as an ideal sensor model (hence, as $P(\overline{D}_u^t|\nu^t, s_u^t) = 0$ in the cells under the UAV at time step t, $\tilde{b}(\nu^t)$ becomes 0 when the UAV overflies cell $\nu^t = s_u^t$).



(a) $b(v^0)$ (b) $\tilde{b}(v^N)$ ${}^*s_{1:U}^{0:N} = \{1,2,5,4,7\}$ (c) $\tilde{b}(v^N)$ $s_{1:U}^{0:N} = \{1,4,5,2,3\}$

Figure 3.8 Illustrative example. (a) Initial belief. (b) Updated "unnormalized belief" corresponding to the optimal trajectory with length $N = 4$ displayed with yellow arrows. (c) Updated "unnormalized belief" corresponding to the trajectory displayed with red arrows.

The optimal trajectory ${}^*s_{1:U}^{1:N}$ (the one with minimum expected time) is displayed with yellow arrows in Figure 3.8 (b). It has been determined after evaluating all the possible trajectories of planning horizon $N = 4$ starting from $s_1^0 = 1$. In this optimal trajectory, first the UAV moves to cell 2 going south, then east to cell 5, north to cell 4, and lastly goes to cell 7. As previously explained, the expected time of a trajectory is computed adding the remaining "unnormalized belief" at each time instant, which is updated using the target dynamic $P(v^t|v^{t-1})$ and sensor $P(\overline{D}_u^t|v^t, s_u^t)$ probability models (Equation 3.23). In this example, as the target is static, $\tilde{b}(v^t)$ is not redistributed and is only modified due to the sensor measurements. Initially, we consider that there is a measurement at the UAV initial location, and hence $\tilde{b}(v^0) = P(\overline{D}_1^0|v^0, s_1^0)P(v^0)$ as Equation 3.16 states, where $P(v^0)$ is the initial probability map. Therefore, the "unnormalized belief" at cell s_1^0 is set to zero (i.e. $\tilde{b}(v^0 = 1) = 0$) and $\sum_{v^0 \in G_\Omega} \tilde{b}(v^0) = 0.9$. Next, at time step $t = 1$, the UAV goes south, $\tilde{b}(v^1 = 2) = 0$ and $\sum_{v^1 \in G_\Omega} \tilde{b}(v^1) = 0.6$. Following the same procedure, in the second time step $\tilde{b}(v^2 = 5) = 0$ and $\sum_{v^2 \in G_\Omega} \tilde{b}(v^2) = 0.4$, in the third time step $\tilde{b}(v^3 = 4) = 0$ and $\sum_{v^3 \in G_\Omega} \tilde{b}(v^3) = 0.25$, and in the fourth time step $t = N$ and $\tilde{b}(v^4 = 7) = 0$ and

$\sum_{v^4 \in G_\Omega} \tilde{b}(v^4) = 0.15$. Hence, the expected time of the optimal trajectory is:

$$ET({}^{*}s_{1:U}^{0:N}) = \sum_{t=0}^{N} \sum_{v^t \in G_\Omega} \tilde{b}(v^t) = 0.9 + 0.6 + 0.4 + 0.25 + 0.15 = 2.3$$

Having an $ET({}^{*}s_{1:U}^{0:N}) = 2.3$ implies that if the UAV realizes infinite times the search following the trajectory ${}^{*}s_{1:U}^{0:N}$ and we compute the mean of the target detection times of all the experiments, we would get a value of 2.3. Note that despite considering discrete time instants the mean does not have to be an integer value. Furthermore, if we consider a basic time step between measurements $\Delta T = 100$ seconds, the expected time would be $100 \cdot ET({}^{*}s_{1:U}^{0:N}) = 230$ seconds.

Figure 3.8 (c) shows in red another possible search trajectory, which is described by the sequence of cells $s_{1:U}^{0:N} = \{1,4,5,2,3\}$, and whose expected time is computed below, after following a similar procedure to the previous one to obtain the intermediate "unnormalized belief".

$$ET(s_{1:U}^{0:N}) = \sum_{t=0}^{N} \sum_{v^t \in G_\Omega} \tilde{b}(v^t) = 0.9 + 0.75 + 0.55 + 0.25 + 0.15 = 2.45$$

As expected, the ET of this second search trajectory is higher than the expected time of the optimal trajectory displayed in Figure 3.8 (b) and therefore, it is a worse trajectory for MTS. The optimal trajectory ${}^{*}s_{1:U}^{0:N}$ is the one with minimum ET for the given search horizon N and initial information (target belief and dynamics, UAV dynamics and sensor model, and UAV initial location) and fulfills

$$\begin{aligned} &\text{minimum} \quad ET(s_{1:U}^{0:N}) = ET({}^{*}s_{1:U}^{0:N}) \\ &\text{subject to} \quad s_u^{t+1} = f(s_u^t, c_u^t) \qquad\qquad u = 1,\ldots,U \quad c_u^t \in \mathbf{C_u} \end{aligned} \tag{3.24}$$

For the illustrative scenario displayed in Figure 3.8, it is possible to evaluate all possible solutions and obtain the optimal one. However, as the problem complexity increases the number of possible solutions rapidly increases too and the computational time required to generate and evaluate all possible solutions makes the

problem resolution intractable. For instance, considering the discrete cardinal UAV dynamical model for a given planning horizon N the number of possible solutions is 8^N (without considering the restricted actions of the cells in the borders of the search area) and increases exponentially with N.

Furthermore, it is worth to mention that the two trajectories displayed in Figure 3.8 have the same final probability of detecting the target ($P_d(s_{1:U}^{0:N}) = 1 - P_{nd}(s_{1:U}^{0:N}) = 1 - 0.15 = 0.85$ and $P_d({}^*s_{1:U}^{0:N}) = 1 - P_{nd}({}^*s_{1:U}^{0:N}) = 1 - 0.15 = 0.85$). Therefore, the strategy of maximizing the probability of detection is not able to distinguish between both trajectories. On the contrary, optimizing $ET(s_{1:U}^{0:N})$ allows us to know that the trajectory displayed with yellow arrows is a better option for MTS as it visits first the cells with higher probability of target presence. Summing up, while maximizing $P_d(s_{1:U}^{0:N})$ only takes into account the total amount of explored belief, for optimizing $ET(s_{1:U}^{0:N})$ not only the total of the probability gathered at the end of the trajectory matters, but also the time order in which the cells with higher probability are visited.

3.5. Probabilistic Search Algorithms

In this section we describe how probabilistic search algorithms solve the search problem. First, we define the main inputs and outputs of the probabilistic search algorithms. Then, we present the receding horizon control approach, commonly used by PS state of the art algorithms, and the myopia problems derived from its use.

3.5.1. Multi-UAV PS Algorithms Input and Output Information

The main objective of Probabilistic Search (PS) algorithms is to propose the best feasible search trajectories according to a probabilistic target detection related criteria taking into account the available uncertain information about the target and sensor performance. In order to meet the mission requirements, PS algorithms should consider, apart from the probabilistic models, other additional information such as the

UAVs dynamic restrictions or non-flying zones. The main PS algorithms inputs and outputs are displayed in Figure 3.9.

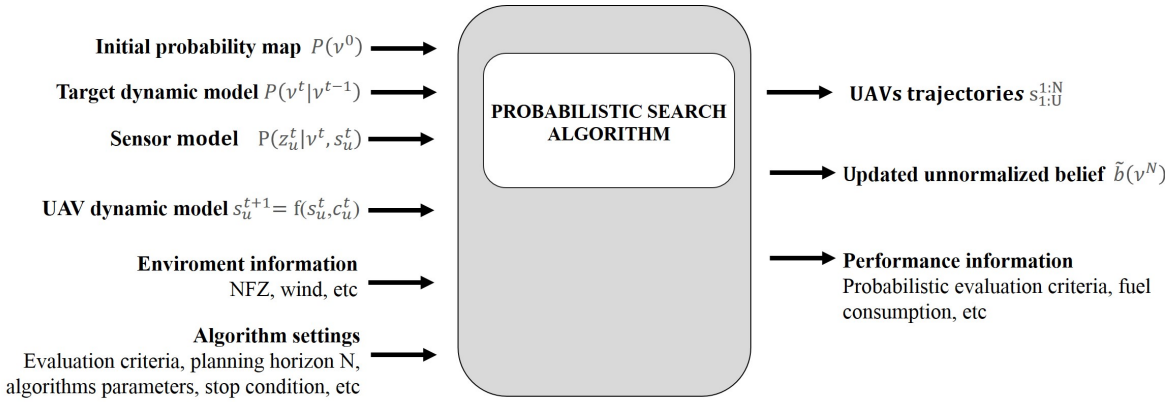


Figure 3.9 Main PS Algorithms input and output information.

The main inputs of PS algorithms are:

Probabilistic models. The uncertain information about the sensor performance and target location and dynamics is modelled with probabilistic functions that are passed as inputs to the PS algorithm. The initial probability map $P(v^0)$ contains the initial information about the target location, generally expressed with a spatial discretization of the target location density function into a rectangular grid. Some PS algorithms allow to include target dynamic information, usually modelled with a Markovian model $P(v^t|v^{t-1})$ that contains the probability that the target moves from its current location cell of the search area to other locations (cells of the search area). The target dynamical model is necessary to update the target belief with the target dynamic information as time passes. Finally, the sensor performance information is contained in the sensor model $P(z_u^t|v^t, s_u^t)$, which returns the probability of measurement z_u^t taking into account the target and UAV states and sensor characteristics (e.g. field of view). The sensor model allows to update that target probability map with new sensor information.

UAVs information. PS algorithms must consider the UAV dynamic information in order to obtain search trajectories that are appropriate for the UAVs from the maneu-

verability point of view. Generally PS algorithms consider as input the UAV dynamic models $s_u^{t+1} = f(s_u^t, c_u^t)$ that allow to obtain the search trajectories from the sequence of optimized control actions $c_{1:U}^{1:N}$ and initial UAV positions $s_{1:U}^0$ (usually considered as a fixed input). Moreover, as generally a bigger number of UAVs obtain better search results, the number of UAVs is usually determined by the maximum number of UAVs available for the search and is considered as a fixed input.

Environment information. Information about the search environment may be also considered. For instance, PS algorithm can consider environment information such as non-flying zones (NFZ) that the UAVs must avoid overflying, the wind direction or speed that can influence in the UAV dynamics, or the weather conditions that can modify the sensor performance.

Algorithm settings. PS algorithms also require information about the settings of the algorithm like the PS evaluation criterion to optimize or the values of the parameters of the algorithm. It is worth noting that the evaluation criterion is the key characteristic that distinguish MTS algorithms from other probabilistic search algorithms. Although MTS algorithms may optimize multiple criteria, the main evaluation criteria should be related to minimize the target detection time (e.g. the expected target detection time). The evaluation criteria allow the algorithm to compare different search trajectories. Besides, the PS algorithms require the length or total time of the search trajectories that will determine the number of control variables to optimize. Finally, PS algorithms also require values of the parameters specific of the optimization techniques (e.g. percentage of mutation in a genetic algorithm) or general parameters of the PS algorithm like the maximum number of algorithm iterations.

The main outputs of PS algorithms are:

UAV optimized search trajectories. The best search trajectories $s_{1:U}^{0:N}$ found by the PS algorithm are returned as solution.

Updated information of target location. The updated belief (either in its "normalized" $b(v^t)$ or "unnormalized" $\tilde{b}(v^t)$ version) with the sensor measurements of $s_{1:U}^{0:N}$ is usually returned as output, as its comparison with the initial $b(v^0)$ allows to easily

determine the areas explored by the UAVs, or the areas to explore in the future if the search carried out following $s_{1:U}^{0:N}$ is unsuccessful.

Performance information. The fitness criteria and other performance information about the proposed solution $s_{1:U}^{0:N}$ can be also returned.

3.5.2. Receding Horizon Control (RHC) Approach

The large search space of PS problems (defined by the combinations of the possible control actions values of each UAV over the entire mission) makes intractable to optimize search trajectories with large planning horizons. Receding Horizon Control (RHC) is a technique commonly used to optimize large planning horizons by iteratively optimizing over a shorter horizon. In this way, the optimization of the full search trajectory $s_{1:U}^{0:N}$ is divided in several steps, where subsequences with a smaller planning horizon L are sequentially optimized. In other words, $s_{1:U}^{0:N}$ is divided and obtained as $[s_{1:U}^{0}, s_{1:U}^{1:L}, s_{1:U}^{L+1:2L}, \ldots, s_{1:U}^{N-L+1:N}]$. More in detail, and as Figure 3.10 shows, during the optimization of the first subsequence the initial belief $P(v^0)$ and UAVs positions $s_{1:U}^{0}$ are considered as inputs, but for the optimization of following subsequences the final UAV positions $s_{1:U}^{qL}$ and the updated "unnormalized belief" $\tilde{b}(v^{qL})$ of the previous optimization step are considered instead. In this way, for the optimization of the full trajectory $s_{1:U}^{0:N}$ with planning horizon N, we would require $Q = N/L$ subsequences, being L the smaller planning horizon of the subsequences.

RHC can be applied both in offline or online planning. In offline planning, the total plan is obtained combining the optimized subsequences of length L and the stop condition of each optimization step (computational time or number of iterations) can be chosen without restrictions. In online planning, the optimization of a subsequence is done during the execution of the previous step. Therefore, the flying time required for executing a subsequence defines the maximum computational time available for the optimization of each subsequence, before the current plan is completed and the cycle is repeated. The stricter computational time restrictions of online approaches

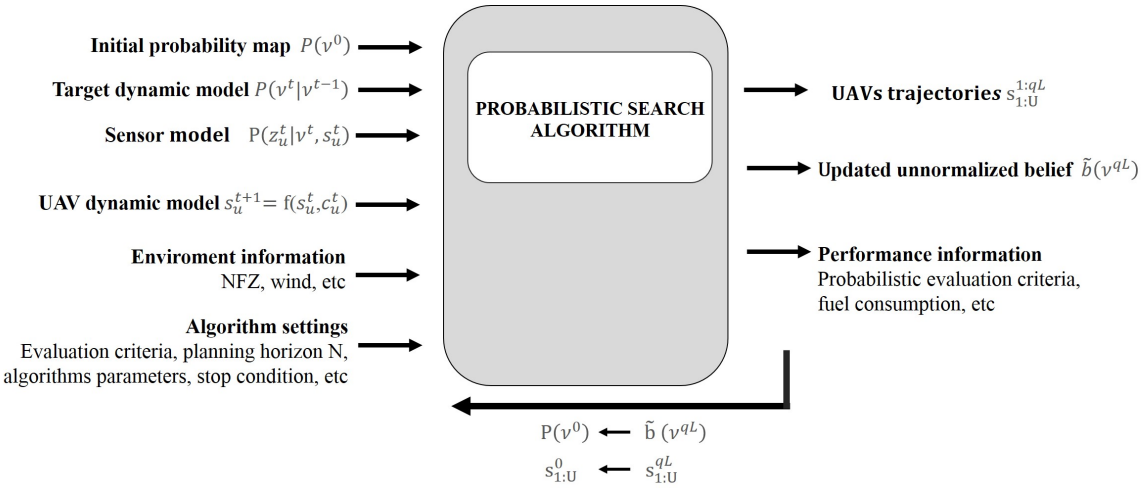


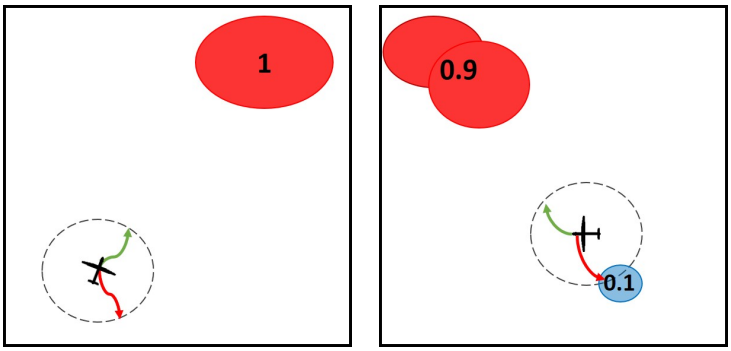
Figure 3.10 Main PS Algorithm with receding horizon control input and output information.

impose to maintain the lookahead depth L very short, making worse the myopia effects derived from RHC.

Summing up, RHC is a widely used technique that allows to obtain an approximate solution with reasonable computational resources at expenses of possibly obtaining myopic solutions due to the limited horizon. Alternatively, in this thesis we propose to incorporate MTS constructive heuristic information to allow to increase the considered planning horizon. In particular, the ACO based methods proposed in this thesis, instead of starting without any knowledge about what are the best actions to take by each UAV, use the information of a MTS constructive heuristic through the optimization process. This allows to build acceptable solutions even from the first algorithm iterations and accelerate the convergence of the algorithms to overall good trajectories, allowing in this way to consider bigger planning horizons. Furthermore, it is worth clarifying that this alternative approach of considering MTS constructive heuristics is not incompatible with the RHC technique, so both techniques can be used in conjunction if the computational restrictions require it.

3.5.3. Myopic Solutions

The RHC method divides the optimization of the problem in several steps allowing to reach a good solution in a feasible computational time. However, each RHC optimization step tries to find the best solution within the given horizon without considering the suitability of the current solution in future optimization steps. Therefore, the limited horizon may cause myopia problems.



(a) (b)

Figure 3.11 Possible myopic situations due to the limited horizon (indicated with a dash line) where (a) the UAV is unable to distinguish between any trajectory and (b) the UAV chooses a myopic solution. Colored areas indicate areas with target probability presence, myopic and non-myopic solutions are respectively displayed with red and green colored arrows.

Figure 3.11 shows two possible myopic situations that may be derived from RHC method. The first situation, illustrated in Figure 3.11 (a), happens when all the regions with target presence probability are separated by a distance much greater than an agent can cover on the horizon of a optimization step. In this case, all possible paths from the current UAV position do not gather any belief and thus would have the same fitness value. Therefore, the algorithm is trapped and unable to realize that although in the current optimization step the gain of all the paths is the same, the solutions that move the UAV closer to the high probability areas (as the green trajectory in Figure 3.11 (a)) are better choices for the following optimization steps. In the second situation, illustrated in Figure 3.11 (b), the algorithm guides the UAV to a high probability area that is within the current horizon instead to the further away

higher probability area outside the horizon reach. Hence, due to the limited horizon, the chosen path (displayed in red in Figure 3.11 (b)) is not the best option for the remaining steps of the trajectory.

As it was already seen in Chapter 2, with the purpose of reducing the myopia effects some works include myopia avoiding strategies. Some myopia strategies only deal with myopic situations as the one shown in Figure 3.11 (a), where the UAV is far away from all high probability areas and thus, the fitness of the possible trajectories within the limited horizon are really similar. For instance, (Tisdale et al., 2009) and (Wong et al., 2005) identify this myopic situation when the fitness value of the best solution is below a certain threshold, and to amend it the former increases the optimization horizon and the later directs the UAV to the closer mode of the belief. There are other type of strategies that can deal with both type of myopic situations of Figure 3.11 by utilizing some function that estimates long term rewards. For instance, the heuristic presented in (Lanillos et al., 2014b) is modelled as a long term sensor, and the one proposed in (Pérez-Carabaza et al., 2016b) weights the possible reachable belief with a function of the distance, giving higher rewards to closer areas.

3.6. Metaheuristics

Due to the high complexity of PS problems, optimal solutions can only be obtained for very simple search scenarios and simplified formulations of the problem (Eagle, 1984). Therefore, PS algorithms are generally based on approximated optimization techniques or metaheuristics (such as several of the examples already reviewed in Chapter 2).

In computer science and mathematical optimization, a metaheuristic is a high-level procedure that can be adapted to a variety of problems, and is able to provide high quality solutions to them. Metaheuristics deal wisely with only a portion of the possible set of solutions of the problem, and thus, although their use does not ensure solution optimality, they generally return high quality solutions. Therefore,

metaheuristics are advantageous for solving high complexity problems where exact optimization methods would require excessive amount of time.

In order to return fast UAV routes to find the target, the MTS algorithms proposed in this thesis use metaheuristics inspired in the mechanisms used by ants to find fast paths to food sources. We have selected Ant Colony based Optimization (ACO) metaheuristics due to 1) their good performance in a variety of problems (Blum, 2005) and 2) their ability to include problem specific information through the use of constructive heuristics.

3.6.1. Introduction to Ant Colony based Algorithms

Ant Colony Optimization (ACO) algorithms are metaheuristics belonging to Swarm Intelligence, the discipline that deals with natural and artificial systems composed of many individuals that are coordinated using decentralized control and self-organization. Swarm intelligence systems typically consist of a population of simple agents which follow simple rules, without a centralized control structure dictating how individual agents should behave and whose "intelligent" global behavior emerges from the indirect interaction of the individual agents through the environment (stigmergy).

ACO algorithms are inspired in the natural swarm intelligence system of ant colonies, where the agents (ants) share information through pheromone deposit. Pheromones are chemicals secreted by individuals that impact the behavior of the receiving individuals of the same species, and in the case of ants, they are used in their foraging activity.

The utility of ant pheromones deposit mechanism was observed in the double bridge experiment presented in (Goss et al., 1989). The experiment, sketched in Figure 3.12, consists of an ant nest that is connected to a food source by a bridge composed of two identical modules that have two branches of different lengths. Either when an ant goes to the food source or returns with food to the nest, it has to choose twice between a short branch and a large branch (in the graphic of Figure 3.12, points 1 and 3 for going to the food source and points 2 and 4 for coming back).

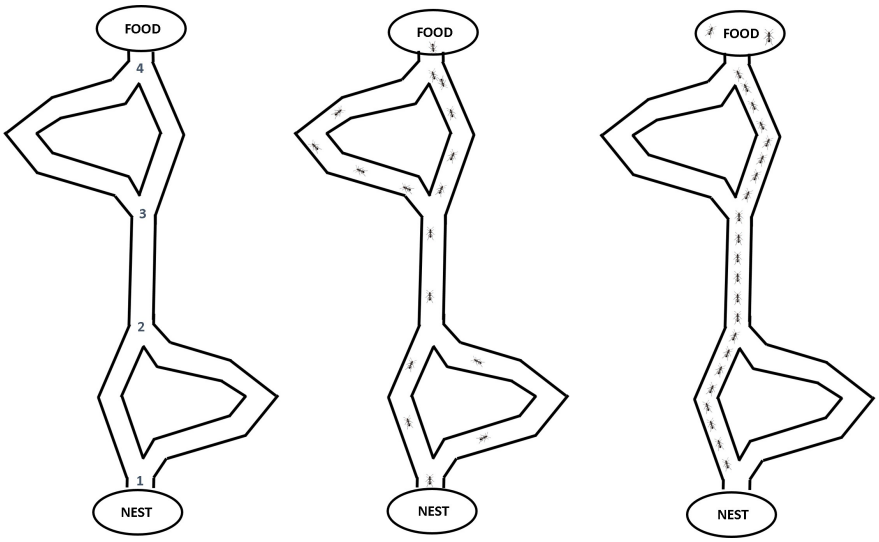


Figure 3.12 From left to right: a double bridge experiment set up, ants distribution at the beginning of the experiment, and ants distribution minutes later.

Initially, ants choose equally all ways, but some minutes later the shorter branches become visibly preferred. The reason that enables ants to find the shorter way to food is the positive reinforcement of pheromones that makes ants to choose preferentially the directions where there is a higher pheromone concentration. For instance, as it takes longer time to complete the larger branch, the first ant that arrives to the food has probably chosen the shorter branch of point 3, and therefore on its way back the shorter branch has higher pheromone concentration. This study is the origin and inspiration of the posteriori proposed ant colony based optimization algorithms.

In 1991 Marco Dorigo proposed in his thesis the first algorithm for solving combinatorial problems inspired in ant colonies, Ant System (AS) (Dorigo et al., 1996), which is a population based iterative algorithm that uses the concept of stimergy to share the information among a population of artificial ants through a pheromone table. Besides, in contrast to natural ants, artificial ants are not blind and consider a constructive greedy heuristic (visibility) to decide each step of their tour.

Dorigo initially applied AS to solve the well known Travelling Salesman Problem (TSP), which aims to find the closed tour with minimal length that visit once a group of cities. For solving TSP, each element of the ant tour (next city j to visit from

current city i) is decided sampling from a probability function that combines the information saved in the pheromone table $\boldsymbol{\tau}$ with a TSP specific heuristic η.

$$P(i,j) = \frac{\tau(i,j)^{\alpha}\eta(i,j)^{\beta}}{\sum_{k \in allowed} \tau(i,k)^{\alpha}\eta(i,k)^{\beta}} \tag{3.25}$$

Equation 3.25 states the probability the probability $P(i,j)$ that an ant chooses going from city i to city j is computed combining the pheromone $\tau(i,j)$ and heuristic $\eta(i,j)$ information considering the parameters α and β that control the pheromone and heuristic influence. The division term ensures that the probability that an ant moves to all possible destination cities is one, $\sum_{k \in allowed} P(i,k) = 1$, where allowed cities are the ones that have not been visited yet. In AS formulation for TSP, the heuristic proposed by Dorigo is inversely proportional to the distance between the cities, giving in this way higher chances (heuristic values) of being selected to the cities that are closer to the current position i of the ant.

At each algorithm iteration, after the M ants have constructed their tours according to Equation 3.25, the pheromone table is updated with the new information. In AS, all ants increase the pheromone values corresponding to their tours with an amount proportional to the fitness (length of the tour in TSP). In this way, the pheromone table is initialized uniformly and does not contain any information, but after the pheromone update process takes place it contains information about what decisions have been more successfully in the previous iterations. Besides, in analogy with the pheromone evaporation process of nature, Dorigo considers the evaporation of pheromone trails controlled by a pheromone evaporation parameter ρ. The pheromone evaporation allows the pheromone table to partially forget previous knowledge and gives more importance to the information learned in near previous iterations. Furthermore, the consideration of problem specific heuristic allows to include problem specific information into the algorithm and helps AS to find acceptable solutions from the early algorithm iterations.

After the publication of AS by (Dorigo et al., 1996), several ant colony based algorithms have been proposed and applied to a big variety of combinatorial problems ranging from quadratic assignment problem (Stützle and Dorigo, 1999) to generating test data for software (Mao et al., 2015). Among the existing ACO algorithms for combinatorial optimization Ant Colony System (ACS) by (Dorigo and Gambardella, 1997) and Max-Min Ant System (MMAS) by (Stützle and H. Hoos, 2000) stand out for being successfully used in different applications. These ACO algorithms maintain the basic idea of AS but implement different pheromone update processes with the intention of avoiding the early stagnation, i.e. the situation when all the ants make the same tour without exploring new possibilities. Regarding this thesis, it is worth noting that the MTS algorithm we propose in Chapter 4 for solving MTS with discrete UAV dynamical models is based on MMAS. We use MMAS to solve our problem because we can benefit from the parallel generation of the ants tours, which is not possible with the ACS variant due to a local pheromone update rule that changes the pheromones after each ant step.

ACO-based algorithms were originally designed to solve complex combinatorial problems where exhaustive methods required intractable calculation times. Their good performance derived in a research interest on extending these techniques to continuous optimization problems. However, in continuous implementations the learned information can no longer be saved in a pheromone table, as the domain of each solution component is no longer a finite set. Among ACO algorithms for continuous domains Ant Colony Algorithm for Continuous Domains (ACOR) proposed by (Socha and Dorigo, 2006) stands out. For this reason, the MTS algorithm we propose in Chapter 5 for solving MTS problems with continuous UAV dynamical models is based on ACOR.

3.7. Summary

This chapter states the MTS problem main objective: obtaining the search trajectories of a fleet of UAVs that will find the target in minimum time considering the

available information about the sensor performance, target location and dynamics, and subject to the UAV dynamical restrictions.

Due to the uncertainty associated to the problem (target location, target dynamics and sensor performance), it is tackled from a probabilistic approach, which allows to update the target location information (with the target dynamics and sensor measurements) through a RBF filter, and to evaluate and compare different search trajectories based on an appropriate probabilistic criteria (such as the probability or expected time of target detection).

All PS algorithms consider as basic inputs the initial target probability map and the sensor performance model and return as output the optimized search trajectories. Besides, depending on the PS algorithm, other input and output information like the target motion model, non-flying zones or fuel consumption can be also considered. Moreover, the selected probabilistic fitness criterion depends on the specific problem tackled by the PS algorithm. In this thesis we select the expected value of the target detection time (ET) due to its adequacy for MTS.

Finally, this chapter introduces ant colony based algorithms, a metaheuristic inspired in the foraging behavior of ants. We have selected this technique for the MTS algorithms proposed in this thesis due to their good performance in a variety of high complexity problems. Besides, the heuristic mechanism of ACO enables us to introduce MTS specific knowledge and obtain high quality solutions in less computational time.

Chapter 4

MTS Algorithms for Cardinal UAV Motion Models

"The five separate fingers are five independent units.
Close them and the fist multiplies strength. This is organization."
James Cash Penney

This chapter proposes the use of Ant Colony Optimization (ACO) techniques for a discrete version of the MTS problem with UAVs moving according to the eight cardinal directions. ACO is a nature inspired metaheuristic that has shown good performance solving combinatorial problems with high computational complexity, such as TSP (Dorigo et al., 1996). It is also an iterative algorithm whose initial population of solutions is improved iteration by iteration using the information of the best found solutions in previous iterations. A distinguishing feature of ACO from other population based algorithms (such as genetic algorithms) is the inclusion of problem specific heuristic information during the solution construction process. In this chapter, we propose a specific MTS heuristic that guides the UAVs toward the higher and closer areas of the target probability map. The main objective of the chapter is to analyze the adequacy of ACO for MTS, and its advantages and disadvantages versus other approaches.

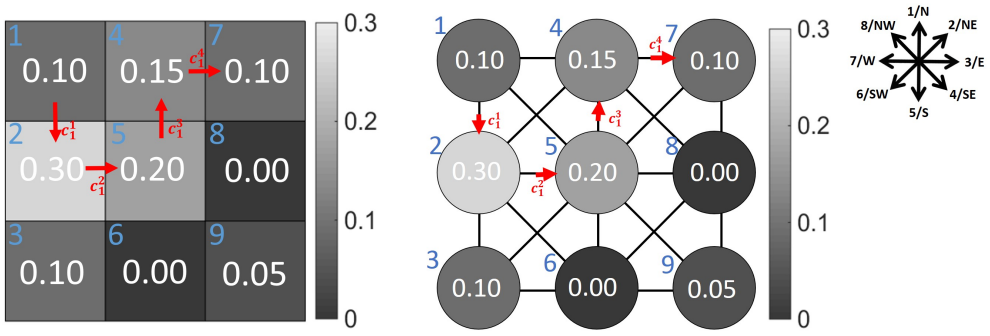
The chapter is organized as follows. First, we present the MTS problem with discrete UAV motion and sensor likelihood models, and describe how the solutions (search trajectories) are codified and evaluated. Next, the proposed MTS algorithm based on Max-Min Ant Colony (MMAS) by (Stützle and H. Hoos, 2000) is introduced. Finally, the performance of the proposed MTS algorithm is analyzed over several search scenarios and compared with other MTS algorithms based on Genetic Algorithm (GA), Cross Entropy Optimization (CEO), Bayesian Optimization Algorithm (BOA) and with three problem specific heuristics proposed in (Meghjani et al., 2016).

4.1. MTS Discrete Approach

In this chapter a complete discrete version of the MTS problem is considered. First, the initial density function about the target location $P(v^0)$ is discretized into a grid G_Ω of w_x x w_y square cells, over which the target movements can be defined between adjacent cells by the target dynamic model $P(v^t|v^{t-1})$. Besides, the UAV states (locations) are also restricted to a constant height and to the centers of the cells of G_Ω, and the movements allowed by the discrete UAV motion model $s_u^{t+1} = f(s_u^t, c_u^t)$ are limited to the centers of neighbor cells. Thereby, search trajectories $s_{1:U}^{1:N}$ can be defined as sequences of overflown cells. Furthermore, as the main objective of the chapter is to analyze the adequacy of ACO for MTS, for simplification purposes, we have selected a sensor ideal model and optimize uniquely the main MTS objective: the expected detection time.

4.1.1. UAV Models

The selected UAV models for the discrete MTS approach are explained below: a simplified UAV motion model and an ideal sensor model.



(a) (b)

Figure 4.1 Simple search scenario with w_x x $w_y = 9$ cells, represented with (a) a grid (b) a graph. The identifying numbers of each cell are shown in blue and the initial probability of target presence in white within each cell. The search trajectory defined by initial cell $s_1^0 = 1$ and cardinal actions $c_1^{1:4} = \{5,3,1,3\}$ is represented with red arrows. The rose compass at the right associates each cardinal direction with an identifying number.

4.1.1.1. UAV Cardinal Motion Model

The mobility of each UAV is discretized in time allowing the UAVs to make decisions at discrete time intervals (time steps). Besides, discrete MTS algorithms also discretize UAVs mobility in space, by limiting the UAVs locations to a constant height and to the centers of the cells of G_Ω, and the UAV movements to the cardinal directions.

The considered cardinal model allows to conceptually move each UAV from its current state s_u^t (cell) to any of adjacent cell following as high level commands the cardinal directions (N, NE, E, SE, S, SW, W, NW). Therefore, as the example of Figure 4.1 (a) shows, the trajectory of a UAV can be equally defined by a sequence of cells $s_1^{0:4} = \{1,2,5,4,7\}$ or by the initial cell $s_1^0 = 1$ and a sequence of high level commands $c_1^{1:4} = \{S,E,N,E\}$, numbered accordingly to the rose compass in Figure 4.1 as $c_1^{1:4} = \{5,3,1,3\}$. The search space and possible movements can be alternatively represented with a graph, as the one shown in Figure 4.1 (b). In this alternative representation, each vertex of the graph represents a cell of the discretized search area and its edges connect adjacent vertexes accordingly with the 8 cardinal directions.

All the cells of G_Ω have associated eight possible actions with the exception of the cells at the borders. Hence, the allowed cardinal actions associated to the cells only depend on the dimensions (w_x, w_y) of G_Ω, which can be calculated at the beginning of the algorithm to use this information to ensure that the algorithm solutions (UAV trajectories) are always kept inside the search area. The cardinal actions associated to all the cells of the example of Figure 4.1 are listed below.

$$s_u^t = 1 \rightarrow \{3,4,5\},\ s_u^t = 2 \rightarrow \{1,2,3,4,5\},\ \ s_u^t = 3 \rightarrow \{1,2,3\},$$
$$s_u^t = 4 \rightarrow \{3,4,5,6,7\},\ s_u^t = 5 \rightarrow \{1,2,3,4,5,6,7,8\},\ s_u^t = 6 \rightarrow \{1,2,3,7,8\}$$
$$s_u^t = 7 \rightarrow \{5,6,7,8\},\ s_u^t = 8 \rightarrow \{1,5,6,7,8\},\ s_u^t = 9 \rightarrow \{1,7,8\}$$

Discrete dynamic models are adequate for rotatory-wing UAVs such as quadrotors. Due to its high maneuverability, these types of UAVs are advantageous in wilderness search and rescue missions, or in search missions in urban environments. Besides, discrete dynamic models have also the advantage of requiring less computational time than continuous ones. This is a great advantage in high complexity problems like MTS. It is worth mentioning that with the considered cardinal model the number of possible solutions roughly grows exponentially with the planning horizon according to 8^N (without considering the lower number of possible actions in the edges). Finally, the MTS discrete algorithm proposed in this chapter can be easily adapted to fixed-wing UAVs by imposing a maximum turning rate, which is the approach taken in several MTS works such as (Raap et al., 2016) or (Yao et al., 2017). For instance, with a maximum turning rate of 45 degrees per UAV displacement, in each cell not in the border 3 actions (turning left, turning right or continue straight) can be applied and translated into the 8 cardinal direction codification taking into account the orientation of the previous UAV displacement.

4.1.1.2. Sensor Model

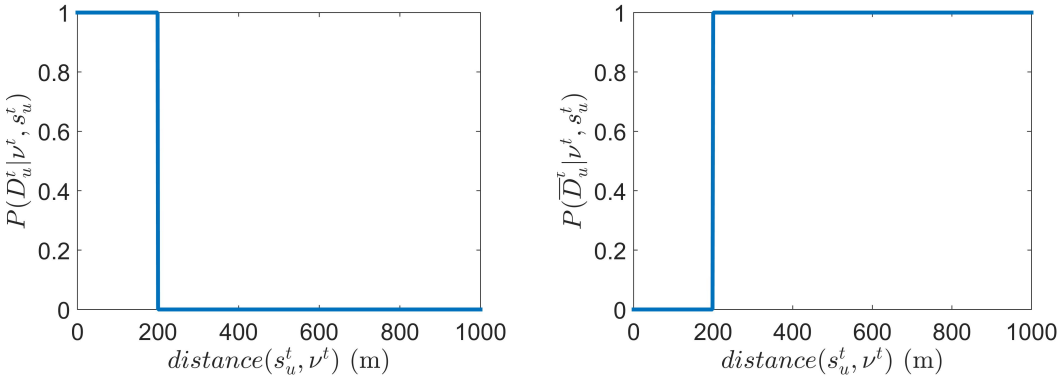
Although the MMAS based algorithms proposed in this chapter can be used with any type of sensor, as our objective is to test the adequacy of ACO techniques for MTS, we have selected the simplified sensor detection function modelled by

Equation 4.1.

$$P(z_u^t = D|\nu^t, s_u^t) = \begin{cases} 1 & \nu^t = s_u^t \\ 0 & \nu^t \neq s_u^t \end{cases} \tag{4.1}$$

The model describes the ideal performance of a sensor that detects with probability 1 a target whose position ν^t (cell) is underneath the sensor position s_u^t. Note that we consider the sensor and UAV location equal, as their deviation is negligible with respect the scenario and target location. Figure 4.2 shows the sensor likelihood in terms of the distance from the sensor/UAV to the target position. When the distance is closer than the lateral size of a cell of the grid (200 meters in the example), $P(D_u^t|\nu^t, s_u^t) = 1$ and its complementary probability $P(\overline{D}_u^t|\nu^t, s_u^t) = 0$. On the contrary, for larger distances $P(D_u^t|\nu^t, s_u^t) = 0$ and $P(\overline{D}_u^t|\nu^t, s_u^t) = 1$.

Finally, we want to emphasize again that other sensor models (e.g. the ones presented in Chapter 5 and Chapter 6) can be used in the formulation of the problem introduced in this chapter. Besides, although the ideal model used in this chapter is easier to implement, its abrupt discontinuity induces frequent rough changes in the ET function, hardening the optimization of MTS.



(a) $P(D_u^t|\nu^t, s_u^t)$ (b) $P(\overline{D}_u^t|\nu^t, s_u^t)$

Figure 4.2 Ideal sensor probability model. (a) Probability of target detection $P(D_u^t|\nu^t, s_u^t)$ and (b) probability of non-detection in terms of the distance from the sensor/UAV to the target position for an scenario with square cells of 200 x 200m.

4.1.2. Discrete MTS Formulation

To be able to optimize MTS with ACO, we need first to codify the solutions (UAV search trajectories) in a appropriate way and then define a MTS criterion to evaluate them.

4.1.2.1. Codification of the Decision Variables

ACO techniques allow to find the best tour in a graph G=(V,E). As we have already seen in Figure 4.1 (b), the discrete approach of MTS can be represented with a graph, where the vertexes V correspond to the cells of G_Ω and the edges E correspond to the cardinal actions that connect adjacent cells. A tour in this graph can be described either by the sequences of visited cells or by the cardinal actions taken and the initial cell. The former is a natural codification for a search trajectory. Besides, when the search is performed by multiple UAVs, the sequences of visited cells associated to each UAV can be concatenated. This codification is advantageous for the evaluation process, as the MTS evaluation criterion generally depends directly on the UAV states. However, not all the combination of values within this codification are valid trajectories, because the UAVs can only move from one cell to their adjacent ones. For that reason, MTS algorithms often use this codification as an intermediate codification required to evaluate the search trajectories.

An alternative codification consists in a sequence of integers, where each point of the sequence represents the high-level commands that should be applied to the UAV to move it from its current cell to the next. Analogously, when the search is performed by multiple UAVs, we can concatenate the sequences of high-level commands associated to each UAV, as Equation 4.2 shows, where each decision variable can take values from $\{1,2,3,4,5,6,7,8\}$. This codification has to be transformed into concatenated sequences of visited cells using the UAV dynamic model $s_u^{t+1} = f(s_u^t, c_u^t)$ before being able to evaluate the codified trajectories. Nevertheless, this transformation is straightforward and does not require much computational time. With this codification the majority of the combination of values constitute valid trajec-

tories (except those sequences that make the UAVs leave the search area). However, MTS ACO-based algorithms can easily incorporate mechanisms to force the UAVs to stay within it.

$$\overbrace{c_1^1, c_1^2, ..., c_1^N}^{\text{UAV 1}}, \overbrace{c_2^1, c_2^2, ..., c_2^N}^{\text{UAV 2}}, ..., \overbrace{c_U^1, c_U^2, ..., c_U^N}^{\text{UAV U}} \tag{4.2}$$

4.1.2.2. Evaluation Criterion

For the MTS discrete approach we choose the expected detection time as the objective function to optimize. The expected detection time $ET(s_{1:U}^{0:N})$ of UAVs trajectories $s_{1:U}^{0:N}$ of planning horizon N, derived in Section 3.4.2, is obtained using Equation 4.3 summing up for each time step the values of the "unnormalized belief" $\tilde{b}(v^t)$ for all the cells of G_Ω.

$$ET(s_{1:U}^{1:N}) = \sum_{t=0}^{N} \sum_{v^t \in G_\Omega} \tilde{b}(v^t) \tag{4.3}$$

The updated "unnormalized belief" $\tilde{b}(v^t)$ at time step t is obtained recursively updating for each time step the target dynamic information through the target dynamic model $P(v^t|v^{t-1})$ with Equation 4.4 and the sensor non-detection measurements through the sensor model $P(D_u^t|v^t, s_u^t)$ with Equation 4.5. The starting updated "unnormalized belief" $\tilde{b}(v^0)$ is initialized by Equation 4.6 with the initial probability map $b(v^0)$ and the initial non-detection measurements. Therefore, the ET of a search trajectory depends on the uncertainty sources of the problem: initial probability map $b(v^0)$, sensor model $P(D_u^t|v^t, s_u^t)$ and target dynamic model $P(v^t|v^{t-1})$.

$$\hat{b}(v^t) = \sum_{v^{t-1} \in G_\Omega} P(v^t|v^{t-1}) \tilde{b}(v^{t-1}) \tag{4.4}$$

$$\tilde{b}(v^t) = \sum_{v^t \in G_\Omega} \prod_{u=1:U} (1 - P(D_u^t|v^t, s_u^t)) \hat{b}(v^t) \tag{4.5}$$

$$\tilde{b}(v^0) = \prod_{u=1:U} P(\overline{D}_u^0 | v^0, s_u^0) b(v^0) \tag{4.6}$$

It is worth noting that although we could have decided to solve a constrained multi-objective version of the MTS (including other objectives such as the fuel consumption, NFZ, etc), by considering only the minimization of the expected time in this chapter, we focus the analysis of the proposed MTS algorithms and the comparison with other methods on the core objective of MTS: minimizing the target detection time. The constrained multi-objective version of the MTS will be presented in Chapter 5.

4.2. MTS-ACO Discrete Approach

This section starts introducing the different ant colony based discrete optimization techniques and justifying the selection of MMAS (Stützle and H. Hoos, 2000) for solving MTS. Next, we describe how MTS is formulated in order to optimize the problem with the selected ACO algorithm. More concretely, we describe how the information is encoded in the pheromone table, the proposed MTS heuristic, the solution construction process and the general pseudocode of the two proposed MTS algorithms based on MMAS.

4.2.1. Discrete ACO Algorithms

ACO is a metaheuristic inspired by the foraging behavior of ants, originally introduced to solve the Traveling Salesman Problem (TSP) and currently used to solve computational problems whose objectives can be formulated as finding good tours through graphs. ACO is an iterative algorithm that constructs the tours of M artificial ants sampling a probability distribution that combines the information of 1) artificial pheromones and 2) a problem-dependent heuristic. On the one hand, the pheromones evolve at each iteration exploiting the information of the best solutions that the algorithm has already identified accordingly to the problem objective function. On the

other hand, the heuristic is the same during all iterations of the algorithm and is defined specifically for each problem to include a priori knowledge about promising regions of the search space.

ACO basic pseudocode is shown in Algorithm 2. The algorithm requires the pheromone α and heuristic β influence parameters and the number of ants M. First, it assigns a starting node to all the ants, e.g. in TSP (finding the best tour that visit once a group of cities) the initial nodes are chosen randomly from all the cities/nodes in the tour. Moreover, the pheromones are initialized uniformly. Within the main loop, at each iteration the tours of the M ants are constructed step by step within the solution construction loop (line 5 to 9) combining the information of the pheromone table $\boldsymbol{\tau}$ with the heuristic information η. Each solution component is sampled from $P(i, j)$, which assigns different probabilities to allowed nodes (line 7). Once the M tours are constructed, the population of ants is evaluated and the global best solution is saved. Then, at the end of each iteration the pheromones are updated; all pheromones trails are evaporated and the trails corresponding to all (or a portion of the best ant tours depending on the ACO version) are reinforced with an amount proportional to the ant fitness (line 13). Finally, once the algorithm stop criterion is fulfilled, ACO returns the best solution found so far.

After the publication of the first ACO algorithm (AS by (Dorigo et al., 1996)) several ant colony based algorithms have been proposed. They mainly differ on the pheromone update rules, although some of them include some modifications of the basic ACO described in Algorithm 2. Among the ACO algorithms for discrete optimization two algorithms stand out for their good performance in a variety of problems: Max-Min Ant System (MMAS) by (Stützle and H. Hoos, 2000) and Ant Colony System (ACS) by (Dorigo and Gambardella, 1997). Both algorithms do the pheromone reinforcement considering only the best found solution of the current iteration or since the start of the algorithm (global best solution). Besides, on one hand, MMAS imposes minimum and maximum pheromone bounds $[\tau_{min}, \tau_{max}]$. This imposition has the purpose of limiting the difference between pheromone trails and thus of avoiding the stagnation of ACO, which occurs when all ants follow the same

Algorithm 2 ACO Basic Metaheuristic

Require: α, β, M

1: *Position each ant at starting node*
2: *Initialize pheromones*
3: **while Stop** $\neq true$ **do**
4: **for** $m = 1 : M$ **do**
5: **while** ant tour $\neq finished$ **do**
6: $i \leftarrow$ *Get current node of the k-th ant*
7: $j \sim P(i,j) = \boldsymbol{\tau}(i,j)^{\alpha}\eta(i,j)^{\beta} / \sum_{l \in allowed} \boldsymbol{\tau}(i,l)^{\alpha}\eta(i,l)^{\beta}$
8: $j \rightarrow$ *Update new solution component to m-th ant tour*
9: **end while**
10: **end for**
11: *Evaluate ant tours*
12: *Update best solution*
13: *Update pheromones*
14: **end while**
15: **return** *best found solution*

paths and thus there is no improvement of the solutions and no exploration of new areas of the search space. On the other hand, ACS uses a different rule to combine pheromone and heuristic information that provides a direct way to balance between exploration and exploitation, and a local pheromone update executed at the end of the solution construction loop (line 5 to 9 of Algorithm 2). The local pheromone update that each ant applies after each step to the last edge traversed prevents a parallel implementation of the construction loop. For this reason, we have selected MMAS to tackle the discrete version of the MTS problem. Moreover, MMAS has shown a good performance in a variety of applications ranging from water distribution systems optimization (Zecchin et al., 2006) to UAV path planning (Hai-bin et al., 2009).

4.2.2. Solving MTS with Max-Min Ant System

In addition to the design of a MTS specific heuristic, there are other aspects to take into account when solving MTS with ACO, which differentiate our MTS-ACO algorithm from the original TSP-ACO implementation (Algorithm 2).

In TSP, as the ant tours are circular (start and end in the same node/city), the initial positions of the ants are set randomly (line 1 of Algorithm 2). However, in MTS the initial positions of the UAVs are given as an input. Thereby, in MTS formulation we set the ants initial positions equal to the UAV initial positions $s^0_{1:U}$. This is a common characteristic of the trajectory optimization problems, where the vehicles initial positions are typically fixed. For instance, (Zhang et al., 2010) proposes an algorithm based on Ant System that minimizes the length and threat exposure of the trajectory of a unique UAV from an initial position to a destiny location, thus the algorithm also sets all the ants initial positions equal to the UAV initial location.

Moreover, in TSP all cities are connected, and ants can choose at each construction step a city among all the cities that have not been visited yet (which are the cities that are not in the tabu list of each ant). However, we do not consider a tabu list to avoid visiting cells because 1) a UAV may need to fly back or cross its past route to access a new area of the search zone and 2) the same cell may have to be revisited to ensure that there is no target in it (in case of using a non ideal sensor or when the target dynamics make the belief move to an already visited area). Besides, in MTS formulation nodes (map cells) are connected according with a grid and the UAVs can only move to neighbor cells following the cardinal directions. Hence, we take advantage of this feature and consider only eight possible movements instead of the movements to all the cells of the grid. In this way, at each construction step of an ant tour, we only have to compute the transition rule for the connections allowed by the eight cardinal actions.

In addition, considering only the movements to the eight neighbor nodes instead to all the nodes allows to reduce the dimensions of the pheromone table. The pheromone table instead of learning the adequacy of the movements from each cell to all the other cells of the grid only considers the movements to the 8 neighbor nodes. We label the MTS algorithm which follows this type of encoding in the pheromone table MMAS-NODE+H, due to its close relation to the TSP node-to-node codification and to its capability of exploiting the information provided by a specific heuristic for MTS.

Furthermore, we propose another type of pheromone encoding that learns the best actions to do at each time step (instead of learning the best actions to do at each node as in MMAS-NODE+H). This alternative is inspired by existing MTS approaches based on estimation distribution algorithms (such as CEO (Lanillos et al., 2012) and BOA (Lanillos et al., 2013)), whose objective is to identify the best actions to perform at each time step. In order to highlight the time encoding of the actions, we label this second encoding MMAS-TIME+H.

4.2.2.1. Pheromones

The pheromone deposit is a positive feedback mechanism that takes place at the end of each iteration and enables the ants to learn the best tours from the trails followed by previous ants.

As we have already introduced, we consider two ACO versions with different pheromone encodings. On on hand, in MMAS-NODE+H, the pheromone table $\boldsymbol{\tau}_{\text{NODE}}$ learns the best actions to perform at each node by each UAV. Hence, $\boldsymbol{\tau}_{\text{NODE}}$ is a 3D matrix of size $(8, wx \cdot wy, U)$, whose elements $\tau_{\text{NODE}}[a,i,u]$ are distributed in rows that correspond to each action (a), columns to each cell of the map (i), and the third dimension to each UAV (u). On the other hand, in MMAS-TIME+H, the pheromone table $\boldsymbol{\tau}_{\text{TIME}}$ learns the best actions to perform at each time step by each UAV. Therefore, $\boldsymbol{\tau}_{\text{TIME}}$ is a 3D matrix of size $(8, N, U)$, whose elements $\tau_{\text{TIME}}[a,t,u]$ are distributed in rows that correspond to each action (a), columns to each time step (t), and the third dimension to each UAV (u).

The pheromone update process of MMAS consists of the three phases (reinforcement, evaporation and bounding) detailed below.

On the one hand, the pheromone reinforcement process increases the pheromone trails corresponding to either the best solution of the iteration (ib) or the global best (gb) solution. To tackle the discrete version of MTS, we have chosen to reinforcement of the best solution of the iteration to avoid early convergence to local optimum. In particular, the pheromone reinforcement is detailed by Equation 4.7 for MMAS-

NODE+H and by Equation 4.8 for MMAS-TIME+H.

$$\tau_{\text{NODE}}[{}^{ib}c_u^t, {}^{ib}s_u^{t-1}, u] \leftarrow \tau_{\text{NODE}}[{}^{ib}c_u^t, {}^{ib}s_u^{t-1}, u] + \frac{1}{ET({}^{ib}s_{1:U}^{0:N})} \tag{4.7}$$

$$\tau_{\text{TIME}}[{}^{ib}c_u^t, t, u] \leftarrow \tau_{\text{TIME}}[{}^{ib}c_u^t, t, u] + \frac{1}{ET({}^{ib}s_{1:U}^{0:N})} \tag{4.8}$$

where ${}^{ib}c_u^t$ and ${}^{ib}s_u^{t-1}$ stand for the action and node (of the u-th UAV at time $t-1$) of the best solution of the current algorithm iteration (ib). Therefore, while in MMAS-NODE+H the actions corresponding to the best solution of the iteration ${}^{ib}c_u^t$ applied at ${}^{ib}s_{1:U}^{0:N}$ are intensified, in MMAS-TIME+H the actions corresponding to the best solution of the iteration ${}^{ib}c_u^t$ that are applied at each time-step ($\forall t = 1:N$) are intensified. In both cases, the intensification factor is inversely proportional to the expected time of the best iteration solution, i.e. there is a higher reinforcement for better (lower) ET values.

On the other hand, and in contrast to the pheromone reinforcement that is only applied to the pheromones entries corresponding to the best solution, the pheromone evaporation described by Equation 4.9 for MMAS-NODE+H and by Equation 4.10 for MMAS-TIME+H is applied to the whole pheromone matrix.

$$\tau_{\text{NODE}}[a, i, u] \leftarrow (1-\rho) \cdot \tau_{\text{NODE}}[a, i, u] \tag{4.9}$$

$$\tau_{\text{TIME}}[a, t, u] \leftarrow (1-\rho) \cdot \tau_{\text{TIME}}[a, t, u] \tag{4.10}$$

where $\rho \in (0,1)$ is the evaporation rate parameter.

Finally, the pheromone bounding, characteristic of MMAS, is described by Equation 4.11 for MMAS-NODE+H and by Equation 4.12 for MMAS-TIME+H.

$$\tau_{\text{NODE}}[a, i, u] \leftarrow max\{\tau_{min}, min\{\tau_{max}, \tau_{\text{NODE}}[a, i, u]\}\} \tag{4.11}$$

$$\tau_{\text{TIME}}[a, t, u] \leftarrow max\{\tau_{min}, min\{\tau_{max}, \tau_{\text{TIME}}[a, t, u]\}\} \tag{4.12}$$

where (τ_{min}, τ_{max}) are the pheromone limits imposed by MMAS to avoid that the pheromone trails of certain choices are significantly higher than those ones of others, a fact that could make all ants choose the same path and could cause MMAS stagnation. According to MMAS (Stützle and H. Hoos, 2000), appropriate maximum and minimum pheromone limits are obtained respectively with Equations 4.13 and 4.14.

$$\tau_{max} = \frac{1}{1-\rho} \frac{1}{ET(^{gb}s^{0:N}_{1:U})} \tag{4.13}$$

$$\tau_{min} = \frac{\tau_{max}(1 - \sqrt[n]{P_{best}})}{(avg-1)\sqrt[n]{P_{best}}} = \frac{\tau_{max}(1 - \sqrt[n]{P_{best}})}{7\ \sqrt[n]{P_{best}}} \tag{4.14}$$

where avg is the average number of options chosen among the ants in each construction step (in our problem $avg = 8$, which is the number of cardinal actions) and P_{best} the probability (significantly higher than 0) that an ant chooses the best found solution once the algorithm has converged (for our problem we set $P_{best} = 0.5$). The pheromone limits depend directly (τ_{max}) or indirectly (τ_{min}) on the fitness of the best solution found by the algorithm $ET(^{gb}s^{0:N}_{1:U})$, thus they are updated every time the algorithm finds a better solution.

Besides, MMAS initializes all the pheromones in such a way that after the first iteration all the values correspond to the maximum pheromone limit. To do it, the pheromones table is initialized with an arbitrarily high value, so after the pheromone bounding step of the first iteration, all values are forced to τ_{max}, obtained with Equation 4.13 (according to the pheromone evaporation parameter and the best fitness value obtained after the first iteration).

In summary, MMAS initializes the pheromone table to τ_{max}, achieving in this way a higher exploration of solutions at the start of the algorithm. And at each iteration, the pheromone trails of the best solution of the iteration are reinforced, all the trails evaporated and kept among the pheromone limits (τ_{min}, τ_{max}).

Furthermore, we take advantage of the knowledge about the actions that lead outside the search area in the edge cells of G_Ω and set some zero values in $\boldsymbol{\tau}_{\text{NODE}}$ to avoid the ants from choosing the forbidden actions that lead the UAVs outside

$$\boldsymbol{\tau}_{\text{NODE}} = \begin{array}{c} \\ \text{actions} \end{array} \overbrace{\begin{bmatrix} 0 & 0.86 & 0.86 & 0 & 0.86 & 0.86 & 0 & 0.86 & 0.86 \\ 0 & 0.86 & 0.86 & 0 & 0.86 & 0.86 & 0 & 0 & 0 \\ 0.86 & 0.86 & 0.86 & 0.86 & 0.86 & 0.86 & 0 & 0 & 0 \\ 0.86 & 0.86 & 0.86 & 0.86 & 0.86 & 0 & 0 & 0 & 0 \\ 0.86 & 0.86 & 0 & 0.86 & 0.86 & 0 & 0.86 & 0.86 & 0 \\ 0 & 0 & 0 & 0.86 & 0.86 & 0 & 0.86 & 0.86 & 0 \\ 0 & 0 & 0 & 0.86 & 0.86 & 0.86 & 0.86 & 0.86 & 0.86 \\ 0 & 0 & 0 & 0 & 0.86 & 0.86 & 0 & 0.86 & 0.86 \end{bmatrix}}^{\text{nodes}}$$

$$\boldsymbol{\tau}_{\text{TIME}} = \begin{array}{c} \\ \text{actions} \end{array} \overbrace{\begin{bmatrix} 0.85 & 0.85 & 0.85 & 0.85 \\ 0.85 & 0.85 & 0.85 & 0.85 \\ 0.85 & 0.85 & 0.85 & 0.85 \\ 0.85 & 0.85 & 0.85 & 0.85 \\ 0.85 & 0.85 & 0.85 & 0.85 \\ 0.85 & 0.85 & 0.85 & 0.85 \\ 0.85 & 0.85 & 0.85 & 0.85 \\ 0.85 & 0.85 & 0.85 & 0.85 \end{bmatrix}}^{\text{time}}$$

Figure 4.3 Initial pheromone tables. More in detail, the non-zero values correspond to $\tau_{max} = 0.86$ obtained with Equation 4.13 for $\rho = 0.5$ and $ET({}^{gb}s_{1:U}^{0:N}) = 2.3$, and to $\tau_{max} = 0.85$ for $\rho = 0.5$ and $ET({}^{gb}s_{1:U}^{0:N}) = 2.35$.

the search zone. In those particular entries of the table, we do not consider the bounding imposed by MMAS. However, in MMAS-TIME+H, as $\boldsymbol{\tau}_{\text{TIME}}$ contains the best actions to apply at each time step, this information can not be contained in the pheromone table. Hence the setting of null values to the forbidden actions has to be made before the sampling process of each construction step.

To show how the information is saved and the best trajectories learned by the pheromone tables $\boldsymbol{\tau}_{\text{NODE}}$ and $\boldsymbol{\tau}_{\text{TIME}}$ we will use again the simple search scenario of Figure 3.8 (with a single UAV searching during 4 time steps in a square region of $w_x \cdot w_y = 9$ cells). The pheromone tables of MMAS-NODE+H and MMAS-TIME+H corresponding to the end of the first and fourth iteration of MMAS-NODE+H and MMAS-TIME+H are respectively displayed in Figures 4.3 and 4.4. Note that in both cases, $\boldsymbol{\tau}_{\text{NODE}}$ is an 8 action x 9 nodes x 1 UAV table, while $\boldsymbol{\tau}_{\text{TIME}}$ is an 8 actions x 4 time steps x 1 UAV table.

Besides, the initial values of the pheromone tables (displayed in Figure 4.3) are set to τ_{max} (with the exception of the forbidden trails of $\boldsymbol{\tau}_{\text{NODE}}$ that lead outside the search area) and thus, the pheromone tables do not contain information about the best trails to follow. The maximum pheromone limit used to initialized the pheromone tables was obtained with Equation 4.13 considering the pheromone evaporation parameter ρ and the best expected time $ET({}^{gb}s_{1:U}^{0:N})$ obtained during the first iteration of the algorithms.

$$\boldsymbol{\tau}_{\text{NODE}} = \begin{array}{c} \text{nodes} \\ \begin{bmatrix} 0 & 0.11 & 0.11 & 0 & \boxed{0.49} & 0.11 & 0 & 0.11 & 0.11 \\ 0 & 0.11 & 0.11 & 0 & 0.11 & 0.11 & 0 & 0 & 0 \\ 0.11 & \boxed{0.49} & 0.11 & \boxed{0.49} & 0.11 & 0.11 & 0 & 0 & 0 \\ 0.11 & 0.11 & 0 & 0.11 & 0.11 & 0 & 0 & 0 & 0 \\ \boxed{0.49} & 0.11 & 0 & 0.11 & 0.11 & 0 & 0.11 & 0.11 & 0 \\ 0 & 0 & 0 & 0.11 & 0.11 & 0 & 0.11 & 0.11 & 0 \\ 0 & 0 & 0 & 0.11 & 0.11 & 0.11 & 0.11 & 0.11 & 0.11 \\ 0 & 0 & 0 & 0 & 0.11 & 0.11 & 0 & 0.11 & 0.11 \end{bmatrix} \end{array} \text{actions}$$

$$\boldsymbol{\tau}_{\text{TIME}} = \begin{array}{c} \text{time steps} \\ \begin{bmatrix} 0.10 & 0.10 & \boxed{0.43} & 0.11 \\ 0.10 & 0.15 & 0.10 & 0.10 \\ 0.10 & \boxed{0.43} & 0.10 & \boxed{0.43} \\ 0.10 & 0.10 & 0.10 & 0.10 \\ \boxed{0.49} & 0.10 & 0.16 & 0.10 \\ 0.10 & 0.10 & 0.10 & 0.16 \\ 0.10 & 0.10 & 0.10 & 0.10 \\ 0.10 & 0.10 & 0.10 & 0.10 \end{bmatrix} \end{array} \text{actions}$$

Figure 4.4 Ending pheromone tables. The highlighted elements show the best action for each node (in $\boldsymbol{\tau}_{\text{NODE}}$) or time (in $\boldsymbol{\tau}_{\text{TIME}}$).

Next, as iterations pass, the pheromone table learns the best trails to follow from the best found tours. The pheromone tables after 4 iterations of the algorithms are displayed in Figure 4.4, where the elements that correspond to the optimal solution are highlighted. In particular, for MMAS-TIME+H the emphasized row in each column of $\boldsymbol{\tau}_{\text{TIME}}$ directly encodes the best action at each time step. Hence, as the best row (action) of the first column (time step) is the 5^{th}, the best action of the second time is the 3^{rd} and so on so forth, the optimal sequence of actions ${}^{gb}c_1^{1:4} = \{5,3,1,3\}$. The interpretation of the encoding for MMAS-NODE+H is more complex, as each element of $\boldsymbol{\tau}_{\text{NODE}}$ encodes which action (row) should be taken at each node (column). In this case, as the initial UAV location $s_1^0 = 1$ and the best value of the first column (node) corresponds to the fifth row (action), the first action to be applied is $c_1^1 = 5$ to make the UAV move to node $s_1^1 = 2$. Next, as the best action (row) for the second node (column) is the third, $c_1^2 = 3$ and $s_1^2 = 2$. Following the same process, we can observe how the final $\boldsymbol{\tau}_{\text{NODE}}$ encodes the sequence of actions ${}^{gb}c_1^{1:4} = \{5,3,1,3\}$ to be taken in each of the nodes of the optimal trajectory ${}^{gb}s_1^{0:4} = \{1,2,5,4,7\}$. The higher values of the elements corresponding to the optimal solution indicate how the algorithms have already learned the optimal path of this simple scenario in just a few iterations.

4.2.2.2. MTS Heuristic

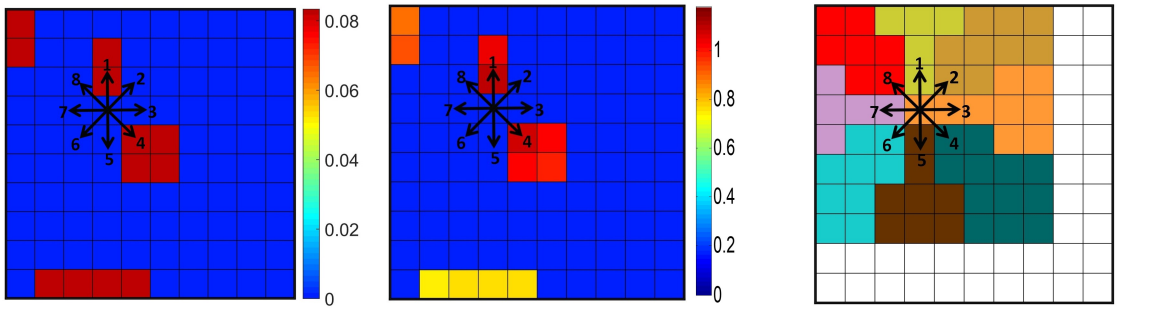
For generating new solutions, ACO combines the information learned in previous iterations with the information given by a problem specific heuristic.

The heuristic we propose for MTS is a spatial function that depends on the current position s_u^t of the ant/UAV and on the current predicted "unnormalized belief" $\tilde{b}(v^t)$. The calculation of the heuristic value for action a of UAV u located at node i at time t, given by Equations 4.15 and 4.16, can be divided in two steps. First, the current predicted "unnormalized belief" $\tilde{b}(v^t)$ is linearly weighted taking into account the distance from the cells of the map j to the current UAV/ant located at cell i with Equation 4.16, giving the highest weight to cell i (where $distance(i,j) = 0$) and lower positive weights to further cells. Besides, the cells that are further from the UAV reach (i.e. $distance(i,j) > N - t$) are not considered during the heuristic calculation (and are assigned a null weight $g(distance(i,j)) = 0$). Second, the heuristic value $\eta(a,i,t)$ associated to action a of the UAV at cell i at time t is obtained adding up the values of the weighted "unnormalized belief" contained in the associated $triangle(a,i,l)$. This triangle is defined by the perpendicular bisector associated to action a starting at cell i and by its length $l = N - t$. In this way, the heuristic function gives higher values to the actions that point toward the highest and closer probability areas. Moreover, it considers only the cells that are reachable from the current UAV location, that is, the cells that are not further that $N - t$.

$$\eta(a,i,t) = \sum_{j \in triangle(a,i,N-t)} g(distance(i,j))\tilde{b}(v^t = j) \tag{4.15}$$

$$g(distance(i,j)) = \begin{cases} N + 1 - distance(i,j) & if \;\; distance(i,j) < N - t \\ 0 & otherwise \end{cases} \tag{4.16}$$

To clarify the heuristic explanation we use the search scenario of w_x x w_y =10 x 10 cells of Figure 4.5 (a), where $\tilde{b}(v^t)$ has four high probability areas and the UAV location $s_u^t = i$ is in the center of the arrows that represent the eight possible actions. Figure 4.5 (b) shows the distance weighted belief, where the further away



(a) $\tilde{b}(v^t)$ (b) Distance weighted $\tilde{b}(v^t)$ (c) $triangle(a, s_u^t, N-t=4)$

Figure 4.5 Heuristic sketch.

probability areas have associated lower weights. Lastly, Figure 4.5 (c) represents in different colors the associated triangles $triangle(a,i,N-t)$ of the eight possible actions. After adding up the weighted probabilities of the cells $j \in triangle(a,i,N-t)$, the highest heuristic value is given to action 4, followed by action 1 and action 8. That is, $\eta(4,s_u^t,t) > \eta(1,s_u^t,t)$ because the triangle of action 4 includes the highest probability zone, and $\eta(1,s_u^t,t) > \eta(8,s_u^t,t)$ because the probability area included in the triangle of action 1 is closer to the UAV position, and thus it has a higher weight. Finally, the heuristic values for the rest of the actions are equal as the probability area at the bottom of the map is out of the reach of the UAV for the given planning horizon N and current time step t, and thus it is not taken into account in the calculation of the heuristic values of any of the cardinal actions.

4.2.2.3. Solutions Construction

At each ACO iteration, M artificial ants obtain the action sequence of each UAV ($c_u^{1:N}$) combining the information of the MTS heuristic and the pheromones.

To do it, the actions of each UAV are sampled incrementally according to the probability rule given by Equation 4.17 for MMAS-NODE+H and by Equation 4.18 for MMAS-TIME+H, which return the probability that action a should be chosen at

time step t for UAV u located at s_u^t.

$$P(a,t,u) = \frac{(\tau_{\text{NODE}}[a,s_u^t,u])^{\alpha}(\eta(a,s_u^t,t))^{\beta}}{\sum_{a=1:8}(\tau_{\text{NODE}}[a,s_u^t,u])^{\alpha}(\eta(a,s_u^t,t))^{\beta}} \tag{4.17}$$

$$P(a,t,u) = \frac{(\tau_{\text{TIME}}[a,t,u])^{\alpha}(\eta(a,s_u^t,t))^{\beta}}{\sum_{a=1:8}(\tau_{\text{TIME}}[a,t,u])^{\alpha}(\eta(a,s_u^t,t))^{\beta}} \tag{4.18}$$

where α and β are parameters that control respectively the pheromone and heuristic influence. Note that, the difference between both equations is due to the fact that while in MMAS-NODE+H the pheromone table $\boldsymbol{\tau}_{\text{NODE}}$ contains the learned information about the best actions to perform at each node, in MMAS-TIME+H the pheromone table $\boldsymbol{\tau}_{\text{TIME}}$ contains the information learned about the best actions to perform at each time step. These probability equations state that is more likely that ants choose the actions with higher pheromones and heuristic values. Hence, on the one hand and due to the pheromone influence, its more likely that a UAV that is at cell i at time t chooses an action that was taken in previous iterations by high quality ant tours at node i (MMAS-NODE+H) or at time step t (MMAS-TIME+H). Besides, on the other hand and due to the heuristic effect, actions that lead the UAVs toward high and close probability areas have higher chances of being chosen.

Finally, note that higher values of α and β increment the pheromone and heuristic influence as they increase the differences between the pheromone or heuristic values associated to different actions. In the extreme cases when $\alpha = 0$ there is no pheromone influence and only the heuristic controls the actions selection process, while when $\beta = 0$ there is no heuristic influence and the ants are guided only by the pheromone information. Setting different values to these parameters will allow us to study effects of the heuristic and pheromones in the proposed algorithms MMAS-NODE+H and MMAS-TIME+H. Besides, it lets us consider two variants without heuristic influence ($\beta = 0$, called MMAS-NODE and MMAS-TIME hereafter) and another one without pheromone influence ($\alpha = 0$, called H from now on).

4.2.2.4. MTS Algorithms based on MMAS

Algorithms 3 and 4 show the pseudocode of the two proposed discrete MTS algorithms: MMAS-NODE+H and MMAS-TIME+H. Both algorithms construct the tours of M artificial ants at each iteration combining the information of the proposed MTS heuristic with the information learned by the previous best ant tours, and are differentiated by the way the learned information is encoded in the pheromone table: best actions taken at each node (in MMAS-NODE+H) or best actions taken at each time step (in MMAS-TIME+H).

In the pseudocodes of Algorithms 3 and 4, we extend our previous notation using bold for the variables corresponding of a whole solution of the population, lower indexes on their right side for indicating the individual/ant (e.g. $\boldsymbol{s}_m$ is the solution $s_{1:U}^{1:N}$ the m-th ant, where $m = 1, ..., M$), and upper indexes at the left to indicate if a solution is the iteration or global best (*ib* or *gb*).

The algorithms inputs (requirements) are the initial positions of the UAVs $s_{1:U}^0$, the number of control actions of the trajectories N, the target initial probability map $P(\mathbf{v}^0)$, the target dynamic model $P(\mathbf{v}^t|\mathbf{v}^{t-1})$, the sensor model $P(z_u^t = \overline{D}|\mathbf{v}^t, s_u^t)$, the UAV dynamic model $s_u^{t+1} = f(s_u^t, c_u^t)$ and MMAS parameters (number of ants M, pheromone influence α, heuristic influence β and evaporation rate ρ).

The algorithms start with the initialization of the global best variables (${}^{gb}\boldsymbol{s}, {}^{gb}\boldsymbol{c}, {}^{gb}\boldsymbol{ET}$) and the pheromone table: $\boldsymbol{\tau}_{\text{NODE}}$ in Algorithm 3 and $\boldsymbol{\tau}_{\text{TIME}}$ in Algorithm 4. The pheromone tables are initialized to an arbitrary high value and after the first iteration to the maximum pheromone limit τ_{max} (Stützle and H. Hoos, 2000). Besides, the values of the pheromones corresponding to the actions that lead the UAVs outside the search area are nullified in $\boldsymbol{\tau}_{\text{NODE}}$.

Within the main iteration loop (between lines 3 up to 27 in both algorithms), M solutions/ants are constructed step by step (between lines 4 and 20) by sampling (in line 13) the combined probability (obtained in line 12) of the information provided by the MTS heuristic function (in line 11, Equations 4.15 and 4.16) and by the pheromone tables. Besides, the "unnormalized belief" $\tilde{b}(\mathbf{v}^t)$ is updated during the

solution construction by each ant with the target motion (in line 8, Equation 4.4) and sensor non detection measurements (in line 16, Equation 4.5), as the value of $\tilde{b}(v^t)$ is required to compute the heuristic function (in line 11). Moreover, in line 17 we take advantage of this calculation to compute the ET of the solutions by iteratively adding up each of the summation terms of Equation 4.3, instead of obtaining the ET once the whole trajectory is available at line 19. In other words, in order to reduce the computational cost of the algorithm, we have interlaced the operations required to construct the solution and compute its ET.

Once the tours of all the ants of the population have been constructed, the algorithms identify and store the best solution obtained within the current population (line 21, *ib*) and so far (line 22, *gb*). Next, the pheromone update process of $\boldsymbol{\tau}_{\text{NODE}}$ in Algorithm 3 or $\boldsymbol{\tau}_{\text{TIME}}$ in Algorithm 4 takes place: the pheromone values corresponding to the iteration best solution (*ib*) are reinforced (in line 23, Equation 4.7 or 4.8), all pheromone values are evaporated (in line 24, Equation 4.9 or 4.10) and bounded (in line 26, Equation 4.11 or 4.12) between the pheromone limits $[\tau_{min}, \tau_{max}]$, computed in line 25 with Equations 4.13 and 4.14. Finally, the planner outputs are the best trajectory found by the algorithm (${}^{gb}\boldsymbol{s}$) and its corresponding expected time of target detection (${}^{gb}\boldsymbol{ET}$).

After explaining the behavior of both algorithms, we want to emphasize the differences between them. In line 2, all the values of the time encoding pheromone table $\boldsymbol{\tau}_{\text{TIME}}$ have to be initialized to the same value, while $\boldsymbol{\tau}_{\text{NODE}}$ has null values in the elements corresponding to the forbidden actions in the border cells. In line 12, the probability distribution of MMAS-TIME+H is obtained with Equation 4.18 and has to be modified to avoid the actions that lead the UAVs outside the search area, while in MMAS-NODE+H the forbidden actions are automatically nullified by their special value in $\boldsymbol{\tau}_{\text{NODE}}$. And finally, in lines 23, 24 and 26, the pheromone reinforcement, evaporation and bounding is done with the corresponding equations of each encoding.

It is worth noting that both algorithms have advantages and disadvantages. On the one hand, at each ant step of MMAS-TIME+H, the actions that lead the UAVs outside

the search region have to be eliminated, while in MMAS-NODE+H we ensure that the trajectories are always valid by nullifying the pheromones corresponding to the actions that will lead the UAVs outside the search region. On the other hand, as the number of actions to optimize N is usually less than the total number of cells $w_x \cdot w_y$ in the search region, $\boldsymbol{\tau}_{\text{TIME}}$ is usually more compact and requires less memory to be stored.

Algorithm 3 MMAS-NODE+H

Require: $N, s^0_{1:U}$ ▷ Number of control actions and initial UAVs locations
Require: $P(\mathcal{V}^0)$, $P(\mathcal{V}^t|\mathcal{V}^{t-1})$, $P(\overline{D}^t|\mathcal{V}^t, s^t_u)$, $s^{t+1}_u = f(s^t_u, c^t_u)$ ▷ Target and UAV models
Require: M, α, β and ρ ▷ ACO parameters
$^{gb}\boldsymbol{ET} \leftarrow \infty, ^{gb}\mathbf{s} \leftarrow [], ^{gb}\mathbf{c} \leftarrow []$ ▷ Initialize fitness function and global best solutions
$\boldsymbol{\tau}_{\text{NODE}} \leftarrow InitializePheromoneNode()$ ▷ Initialize pheromone table
while no finished **do** ▷ Main iteration loop
  for m=1:M **do** ▷ Loop for each ant in the population
    $\tilde{b}(\mathcal{V}^0) \leftarrow \prod_{u=1:U} P(\overline{D}^0_u|\mathcal{V}^0, s^0_u) P(\mathcal{V}^0)$ ▷ Initialize the "unnormalized belief", Eq. 4.6
    $ET(s^0_{1:U}) \leftarrow \sum_{\mathcal{V}^t \in G_\Omega} \tilde{b}(\mathcal{V}^0)$ ▷ ET due to the initial location of the UAVs
    for t=1:N **do** ▷ Solution construction loop
      $\bar{b}(\mathcal{V}^t) = \sum_{\mathcal{V}^{t-1} \in G_\Omega} P(\mathcal{V}^t|\mathcal{V}^{t-1}) \tilde{b}(\mathcal{V}^{t-1})$ ▷ Predict $\bar{b}(\mathcal{V}^t)$, Eq.4.4
      for u=1:U **do** ▷ Loop for each UAV
        $i \leftarrow s^{t-1}_u \in s^{0:t-1}_{1:U}$ ▷ Get the current location (cell) of the UAV
        $\eta(a,i,t) = \sum_{j \in triangle(a,i,N-t)} g(distance(i,j)) \bar{b}(\mathcal{V}^t = j)$ ▷ Eq. 4.15
        $P(a,t,u) = \frac{(\tau_{\text{NODE}}[a, s^t_u, u])^\alpha (\eta(a, s^t_u, t))^\beta}{\sum_{a=1:8} (\tau_{\text{NODE}}[a, s^t_u, u])^\alpha (\eta(a, s^t_u, t))^\beta}$ ▷ Eq. 4.17
        $c^t_u \sim P(a,t,u)$ ▷ Sample the cardinal action to move to the following cell
        $s^t_u \leftarrow f(s^{t-1}_u, c^t_u)$ ▷ Simulate UAV motion model
      end for
      $\tilde{b}(\mathcal{V}^t) = \prod_{u=1:U} P(z^t_u = \overline{D}|\mathcal{V}^t, s^t_u) \bar{b}(\mathcal{V}^t)$ ▷ Update $\tilde{b}(\mathcal{V}^t)$, Eq. 4.5
      $ET(s^{0:t}_{1:U}) \leftarrow ET(s^{0:t-1}_{1:U}) + \sum_{\mathcal{V}^t \in G_\Omega} \tilde{b}(\mathcal{V}^t)$ ▷ Update ET calculation, Eq. 4.3
    end for
    $\boldsymbol{s}_m \leftarrow s^{0:N}_{1:U}, \boldsymbol{c}_m \leftarrow c^{1:N}_{1:U}, \boldsymbol{ET}_m \leftarrow ET(s^{0:N}_{1:U})$ ▷ Store current ant information
  end for
  $[^{ib}\boldsymbol{ET}, ^{ib}\boldsymbol{s}, ^{ib}\boldsymbol{c}] \leftarrow SelectBest(\boldsymbol{ET}_{1:M}, \boldsymbol{s}_{1:M}, \boldsymbol{c}_{1:M})$ ▷ Iteration best solution
  $[^{gb}\boldsymbol{ET}, ^{gb}\boldsymbol{s}, ^{gb}\boldsymbol{c}] \leftarrow SelectBest([^{gb}\boldsymbol{ET}, ^{ib}\boldsymbol{ET}], [^{gb}\boldsymbol{s}, ^{ib}\boldsymbol{s}], [^{gb}\boldsymbol{c}, ^{ib}\boldsymbol{c}])$ ▷ Global best solution
  $\tau_{\text{NODE}}[^{ib}c^t_u, ^{ib}s^{t-1}_u, u] \leftarrow \tau_{\text{NODE}}[^{ib}c^t_u, ^{ib}s^{t-1}_u, u] + 1/^{ib}\boldsymbol{ET}$ ▷ τ_{NODE} reinforcement, Eq.4.7
  $\tau_{\text{NODE}}[a,i,u] \leftarrow (1-\rho) \cdot \tau_{\text{NODE}}[a,i,u]$ ▷ τ_{NODE} evaporation, Eq. 4.9
  $[\tau_{min}, \tau_{max}] \leftarrow PheromoneLimits(^{gb}\boldsymbol{ET})$ ▷ Obtain pheromone limits, Eqs. 4.13, 4.14
  $\tau_{\text{NODE}} \leftarrow max\{\tau_{min}, min\{\tau_{max}, \tau_{\text{NODE}}\}\}$ ▷ Keep τ_{NODE} within allowed range, Eq. 4.12
end while
return $^{gb}\boldsymbol{s}, ^{gb}\boldsymbol{ET}$ (solution with minimum ET)

Algorithm 4 MMAS-TIME+H

Require: $N, s^0_{1:U}$ ▷ Number of control actions and initial UAVs locations
Require: $P(\nu^0)$, $P(\nu^t|\nu^{t-1})$, $P(\overline{D}^t|\nu^t, s^t_u)$, $s^{t+1}_u = f(s^t_u, c^t_u)$ ▷ Target and UAV models
Require: M, α, β and ρ ▷ ACO parameters
$^{gb}\boldsymbol{ET} \leftarrow \infty, ^{gb}\mathbf{s} \leftarrow [], ^{gb}\mathbf{c} \leftarrow []$ ▷ Initialize fitness function and global best solutions
$\boldsymbol{\tau}_{\text{TIME}} \leftarrow InitializePheromoneTime()$ ▷ Initialize pheromone table
while no finished **do** ▷ Main iteration loop
  for m=1:M **do** ▷ Loop for each ant in the population
    $\tilde{b}(\nu^0) \leftarrow \prod_{u=1:U} P(\overline{D}^0_u|\nu^0, s^0_u)P(\nu^0)$ ▷ Initialize the "unnormalized belief", 4.6
    $ET(s^0_{1:U}) \leftarrow \sum_{\nu^t \in G_\Omega} \tilde{b}(\nu^0)$ ▷ ET due to the initial location of the UAVs
    for t=1:N **do** ▷ Solution construction loop
      $\bar{b}(\nu^t) = \sum_{\nu^{t-1} \in G_\Omega} P(\nu^t|\nu^{t-1})\tilde{b}(\nu^{t-1})$ ▷ Predict Predict $\bar{b}(\nu^t)$, Eq.4.4
      for u=1:U **do** ▷ Loop for each UAV
        $i \leftarrow s^{t-1}_u \in s^{0:t-1}_{1:U}$ ▷ Get the current location (cell) of the UAV
        $\eta(a,i,t) = \sum_{j \in triangle(a,i,N-t)} g(distance(i,j))\bar{b}(\nu^t = j)$ ▷ Eq. 4.15
        $P(a,t,u) = \frac{(\tau_{\text{TIME}}[a,t,u])^\alpha (\eta(a,s^t_u,t))^\beta}{\sum_{a=1:8} (\tau_{\text{TIME}}[a,t,u])^\alpha (\eta(a,s^t_u,t))^\beta}$ ▷ Eq. 4.18
        $c^t_u \sim P(a,t,u)$ ▷ Sample the cardinal action to move to the following cell
        $s^t_u \leftarrow f(s^{t-1}_u, c^t_u)$ ▷ Simulate UAV motion model
      end for
      $\tilde{b}(\nu^t) = \prod_{u=1:U} P(z^t_u = \overline{D}|\nu^t, s^t_u)\bar{b}(\nu^t)$ ▷ Update $\tilde{b}(\nu^t)$, Eq. 4.5
      $ET(s^{0:t}_{1:U}) \leftarrow ET(s^{0:t-1}_{1:U}) + \sum_{\nu^t \in G_\Omega} \tilde{b}(\nu^t)$ ▷ Update ET calculation, Eq. 4.3
    end for
    $\boldsymbol{s}_m \leftarrow s^{0:N}_{1:U}$, $\boldsymbol{c}_m \leftarrow c^{1:N}_{1:U}$, $\boldsymbol{ET}_m \leftarrow ET(s^{0:N}_{1:U})$ ▷ Store current ant information
  end for
  $[^{ib}\boldsymbol{ET}, ^{ib}\boldsymbol{s}, ^{ib}\boldsymbol{c}] \leftarrow SelectBest(\boldsymbol{ET}_{1:M}, \boldsymbol{s}_{1:M}, \boldsymbol{c}_{1:M})$ ▷ Iteration best solution
  $[^{gb}\boldsymbol{ET}, ^{gb}\boldsymbol{s}, ^{gb}\boldsymbol{c}] \leftarrow SelectBest([^{gb}\boldsymbol{ET}, ^{ib}\boldsymbol{ET}], [^{gb}\boldsymbol{s}, ^{ib}\boldsymbol{s}], [^{gb}\boldsymbol{c}, ^{ib}\boldsymbol{c}])$ ▷ Global best solution
  $\tau_{\text{TIME}}[^{ib}c^t_u, t, u] \leftarrow \tau_{\text{TIME}}[^{ib}c^t_u, t, u] + 1/ET(^{ib}s^{0:N}_{1:U})$ ▷ τ_{TIME} reinforcement, Eq. 4.8
  $\tau_{\text{TIME}}[a,t,u] \leftarrow (1-\rho) \cdot \tau_{\text{TIME}}[a,t,u]$ ▷ τ_{TIME} evaporation, Eq. 4.10
  $[\tau_{min}, \tau_{max}] \leftarrow PheromoneLimits(^{gb}\boldsymbol{ET})$ ▷ Obtain pheromone limits, Eqs. 4.13, 4.14
  $\tau_{\text{TIME}} \leftarrow max\{\tau_{min}, min\{\tau_{max}, \tau_{\text{TIME}}\}\}$ ▷ Keep τ_{TIME} within allowed range, Eq. 4.12
end while
return $^{gb}\boldsymbol{s}, ^{gb}\boldsymbol{ET}$ (solution with minimum ET)

4.3. Results

In this section we analyze the performance of the proposed MTS algorithm based on MMAS. To do it, we use several search scenarios and an appropriate evaluation methodology explained at the beginning of the section. Then, we analyze the performance of different parameter settings, of the two proposed encodings and of the proposed MTS heuristic. Finally, we compare our MTS algorithm with other heuristic approaches and optimization methods that can be found in the literature.

4.3.1. Scenarios

We analyze the results of the proposed MMAS based approach over several search scenarios presented in Figure 4.6. These scenarios have been previously used in (Lanillos et al., 2013) for showing the capability of a MTS algorithm based on BOA and in (Pérez-Carabaza et al., 2018) for testing the proposed MMAS based MTS algorithm. All scenarios have the same grid size ($w_x = w_y = 21$) and differ in the initial probability map $b(\mathbf{v}^0)$, target motion model $P(\mathbf{v}^t|\mathbf{v}^{t-1})$, number of steps of the horizon N, number of UAVs U and their initial locations $s^0_{1:U}$. The initial beliefs about the target presence are represented with colored maps in Figure 4.6, where cells with higher probabilities of target presence are represented with warmer colors. Besides, the planning horizon N and number of UAVs U are indicated in the top labels of each scenario and the UAVs locations are represented with gray circles over the beliefs. Finally, in case of dynamic targets (indicated in the top labels) the tendency of the target dynamics is sketched with orange arrows over $b(\mathbf{v}^0)$. Further details about each scenario are detailed below.

- **Scenario A** has two high probability areas equally spaced from the UAV initial position. Its difficulty is due to the small difference in ET of the solutions that go to each of the probability zones, as the probability of the area situated in the south is only slightly higher.

- **Scenario B** has the belief initially concentrated in the center of the search area and as time passes the probability moves toward the southeast. The UAV has to intercept and gather the probability mass as soon as possible.
- **Scenario C** complexity lies on the circular spreading movements of two initially concentrated probability areas. Each of the two UAVs in this setup, placed initially at the same location at the center of G_Ω, should follow and gather one of the two probability masses.
- **Scenario D** has a complex target dynamic model that simulates the movements of a lost boat in the sea, which is obtained from a probabilistic wind map. The two central UAVs are expected to intercept the belief and the bottom left UAV to follow the target displacements.
- **Scenario E** initial belief is concentrated in the center of the area and as time passes it spreads out toward the same initial position of the two UAVs, which should first move toward the belief and then turn back to overfly the remaining probability.
- **Scenario F** has two static high probability areas on each side of a single UAV. The UAV needs to go east first, toward the highest probability area, and then fly back over the same trail, toward the other probability zone.

4.3.2. Comparison Methodology

Due to the stochastic nature of the proposed MTS algorithm and of several of the approaches analyzed in this chapter it is necessary a statistical analysis of their results (Besada-Portas et al., 2013). Therefore, we run each algorithm multiple times ($NR = 20$) for each scenario and store the computation time[1] and the solutions with

[1] All algorithms are implemented in MATLAB and run over a 2.81 GHz Intel Core i7 with 8GB RAM PC with Windows 10. Besides, the operations used to evaluate the ET of the solutions are speed up using the MATLAB Parallel Toolbox.

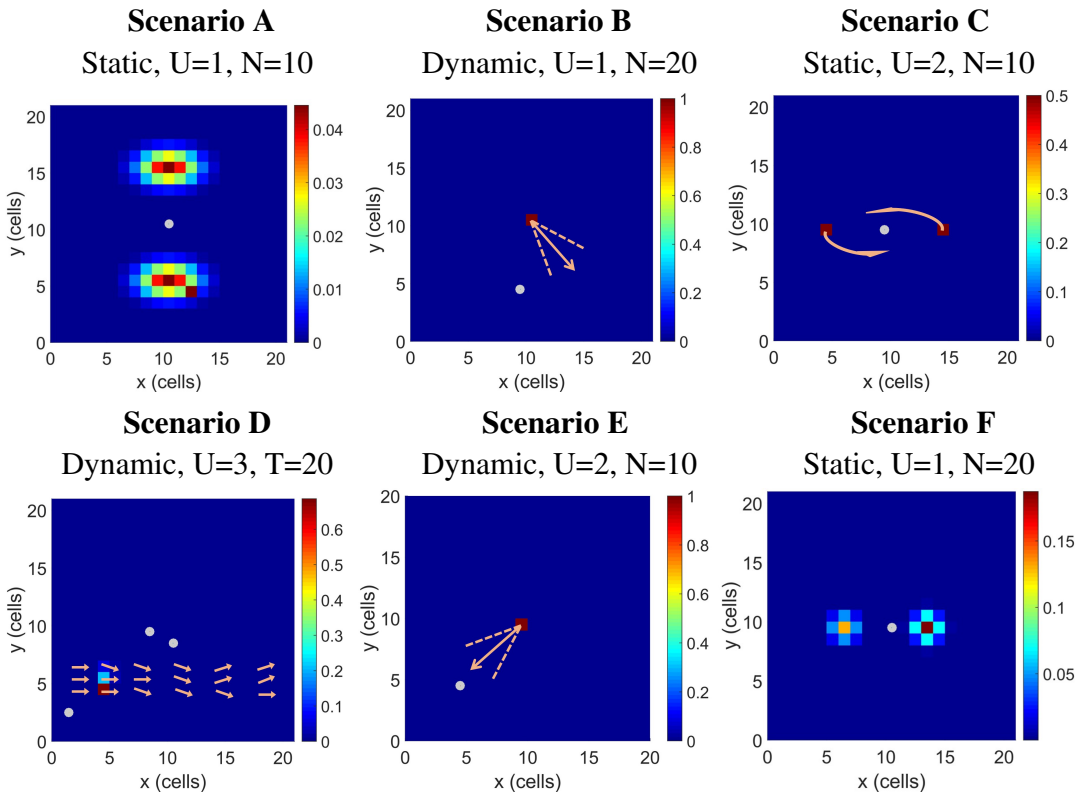


Figure 4.6 Search scenarios, initial probability maps represented with colored matrix, UAV initial states with grey circles and target dynamics with orange arrows.

the best ET ($^{gb}\boldsymbol{ET}$) obtained at the end of each algorithm iteration. We use this information to calculate the mean computation time and ET of each algorithm iteration and to construct two types of comparative graphs explained below and represented in Figures 4.7, 4.8, 4.9, 4.12 and 4.17, where the corresponding graphics to each scenario are sorted in different columns and separated with a horizontal line.

The *dominance evolution graphs* show the dominance relationship between the different approaches along the algorithms iterations. The dominance graphs of the different algorithms/variants analyzed are shown in the first and fourth rows of Figure 4.9 and on the first and third rows of Figures 4.7, 4.8, 4.12 and 4.17. In order to determine if the results are statistically different, we apply the Wilcoxon test (with a significance level of 5%) to compare the results obtained by a base algorithm with the results obtained by the different algorithms/variants at the mean computation time of each iteration. More concretely, the comparison is performed with the

results obtained at the mean computation time of each iteration of the base algorithm against the results obtained by the iteration with the closest (equal or small) mean computation time of the other algorithms. We represent the outcomes of the comparisons in the dominance evolution graphs, displaying the comparison results of each algorithm/variant in different rows (whose labels indicate the algorithm analyzed) at the different computation times of the base algorithm (x-axis) using the following colors: green if the approach indicated in the y-axis dominates the base algorithm, red if the base algorithm dominates the approach indicated in the y-axis, gray if there is not statistical difference, and black when the method in the y-axis has still not finished when the first iteration of the base algorithm has ended. Therefore, the results of the iterations of the algorithms in green are statistically better than the results of the base algorithm, in red statistically worse and in gray similar. Note that, as one algorithm/variant can not dominate itself, the corresponding row of the base algorithm is always completely gray. Besides, as well as indicating the algorithm chosen as the base in the captions of the figures, the base algorithm is always represented in the first row of all the dominance graphs.

The *ET evolution graphs* show the evolution of the fitness criterion (ET) along the algorithms iterations. To construct this type of graph, we use the stored computation times and best ET values of each algorithm iteration to calculate the mean computation time, mean ET and ET standard deviation of each iteration. This information is presented in the ET evolution graphs that appear in the last two rows corresponding to each scenario of Figure 4.9 and in the second and fourth rows of Figures 4.7, 4.8, 4.12 and 4.17. The ET evolution graphs represent with different colors for each algorithm/variant its mean ET (colored line) and standard deviation (delimited by a colored shadowed area) against their computation time. Therefore, those algorithms/variants that show lower ET mean values sooner, converge to a better solution quicker. Besides, over the mean ET curve, we mark with dots the mean computation times of each algorithm/variant every 10 iterations. Hence, the reader can 1) determine the computation time required by 10 iterations of an algorithm/variant by observing the computation time difference of two consecutive dots of its ET evolu-

tion curve and 2) compare the scenarios computation costs based on the density of these dots.

Moreover, as described in Section 4.1.2.2, it is worth mentioning that in both graphs we consider the expected time instant of target detection (the expected time when $\Delta T = 1$). In order to obtain the corresponding ET in time units we only have to scale the $ET(s_{1:U}^{0:N})$ obtained by Equation 4.3 by the considered measurement time lag ΔT, as it was described in Section 3.4.2.

Additionally, it is worth noting that both types of graphics complement each other. On the one hand, the dominance graphs show at the computation time of the base algorithm/variant if there is a statistical difference between the algorithms/variants under analysis, but they do not show how different they are. On the other hand, the ET evolution charts graphically represent the magnitude of the ET difference against the computation time, but they lack of the statistical significance information of the dominance evolution graphs.

We also want to stress that the previous comparison methodology, which combines dominance with mean criterion evolution graphs has been successfully applied in different UAV planning approaches (Besada-Portas et al., 2013; Lanillos et al., 2013; Yang et al., 2015). Moreover, the dominance graphs show, for a given significance level, if the result obtained in a given scenario by one execution of the base algorithm is expected to be statistically better than the results obtained in the same scenario by an execution of any other algorithm in the batch. In other words, we perform an algorithmic pair-wise compassion per scenario, since we want to determine if the base algorithm is a competitive solution against any other algorithm or variant by itself under different MTS settings.

An alternative analysis could be the methodology proposed by (Derrac et al., 2011), which aggregates the comparison results to determine if an algorithm is better than a group of algorithms over a group of scenarios. More in detail, it determines if the final mean result of the runs of an algorithm (and not the results of the different runs at the different computation times as in our case) is better than the mean final result of a group of algorithms (performing a 1-against-N comparison instead of a

pairwise one as in our case) over all the scenarios under test simultaneously (instead of over each one). However, this way of proceeding has two drawbacks. On one hand, it is harder, from a statistical point of view, to conclude under this setup than one algorithm is better than all the others for all the scenarios under test. On the other one, it loses details that are captured by our multi-perspective analysis. In short, our in-deep approach analyses in detail what happens for each scenario, pair of algorithms, and algorithms runs, while the approach in (Derrac et al., 2011) summarizes what happens with the mean of the results over all the scenarios of one algorithm against all the others.

4.3.3. MTS-MMAS Performance Analysis

This section analyzes the performance of the proposed MTS algorithm based on MMAS with MTS heuristic and focuses on the effects of the algorithms parameters, the selected encoding and the use of the ACO pheromones and of the MTS heuristic. First, we analyze the performance of different parameterizations of the proposed MMAS based algorithm and select an appropriate one for the rest of the simulations. Then, we compare the performance of the different pheromone encodings, analyze the power of the proposed MTS heuristic and pheromone learning mechanism and display representative solutions of each scenario.

4.3.3.1. Configuration of MMAS Parameters

The best set of values of ACO parameters depends on the given problem, its heuristic and the available computational time. In general the change of the parameters of an optimization algorithm either increases the exploration of the search space or the exploitation of the learned information during the search process (diversification versus intensification). On the one hand, while further exploration of the search space can derive in finding better solutions, the algorithm usually requires more time to converge to a solution. On the other hand, when the exploitation of the learned information is reinforced, the algorithms tend to converge faster to local solutions.

In the case of MMAS algorithm the parameters that need to be set are M, α, β, ρ and $[\tau_{min}, \tau_{max}]$. The number of ants (M) determines the population size of each algorithm iteration and in general bigger M values lead to a further exploration of the search space, and too small M values can lead to low quality solutions. Besides, α and β respectively control the pheromone and heuristic influence. While high values of α give higher influence to the learned information (exploitation) and may increase the probability of falling into a local optimum, higher values of β intensifies the use of the heuristic information in comparison with the pheromone information accumulated so far. The pheromone evaporation parameter $\rho \in [0,1]$ controls how fast the pheromone values are evaporated (i.e. how quickly the learned information is forgotten). While high values of ρ result in quickly forgetting past knowledge (and may lead the algorithm to be trapped in a local optimum), low values of ρ make the algorithm take a long time before the accumulated knowledge has enough influence in the solution construction process. Finally, the pheromone limits $[\tau_{min}, \tau_{max}]$ help avoiding stagnation that may occur when too high pheromone values lead to all ants choosing the same path. For a more detailed description and analysis of MMAS parameters the reader is referred to (Gaertner and Clark, 2005; Gentile, 2015; Nallaperuma et al., 2015; Pellegrini et al., 2006).

Part of the parameters are automatically set in the MMAS-based algorithms analyzed in this chapter. Similarly to the MTS algorithms presented in (Lanillos et al., 2013) and (Pérez-Carabaza et al., 2018), our MMAS-based approaches have bigger population sizes (number of ants, M) for more complex scenarios. More concretely, the number of ants M is directly proportional to the planning horizon N, the number of UAVs U and the number of possible actions (i.e. the population size is $M = 8 \cdot N \cdot U$). Besides, for setting the pheromone limits $[\tau_{min}, \tau_{max}]$ we follow the procedure proposed in (Stützle and H. Hoos, 2000). Hence, we only have to test the influence of the remaining three parameters (α, β and ρ). To do it, we set up 18 different configurations of each MMAS-based algorithm, varying the parameters within the range of values that can be typically found in the literature: $\rho = \{0.5, 0.1, 0.02\}$, $\alpha = \{1,2\}$, $\beta = \{1,2,3\}$ (Dorigo et al., 1996; Gentile, 2015; Stützle and H. Hoos,

2000). Finally, it is worth mentioning that the analysis presented in this section is not intended to find the best possible set of parameters, but instead a set of parameters that provides high quality MTS solutions in a reasonable computation time.

The values of the parameters of each of the setup configurations of both MMAS-based algorithms and the labels used to identify them (numbers ranging from 1 to 18) are shown in each of the columns of Table 4.1. Figures 4.7 and 4.8 respectively show the dominance and ET evolution graphs of the different configurations of MMAS-NODE+H and MMAS-TIME+H over the six search scenarios. We have selected Configuration 1 as the base algorithm of the dominance evolution graphs, because, except for Scenario A, it shows better (red) or equal (gray) performance than the majority of the remaining configurations. To simplify the ET evolution graphs, we only represent four configurations (1, 2, 7 and 10) that show similar performance to the base configuration and are usually better than the others. Within $\rho = 0.5$, Configuration 2 converges quicker than Configuration 1 in some scenarios, but Configuration 1 is often equivalent or better than Configuration 2 afterwards. Regarding the configurations with lower evaporation rate (ρ = 0.1), Configuration 10 converges quicker to good solutions than Configuration 7, although Configuration 7 reaches better solutions in some scenarios. These facts show how the balance (obtained through the combined values of α, β, and ρ) is achieved for the MTS algorithms: when the evaporation rate is higher (ρ = 0.5), the values of the pheromones and heuristic influence parameters should be low and equally considered ($\alpha = 1$, $\beta = 1$); when the evaporation rate is lower ($\rho = 0.1$) the pheromones importance (exploitation) should be increased ($\alpha = 2$, $\beta = 1$) for a faster convergence. Finally, due to the overall better behavior of Configuration 1 to good solutions, we will select its parameterization for both MMAS encodings.

4.3.3.2. Encoding and Heuristic Analysis

After having selected the algorithms parameters settings (Configuration 1 with $\alpha = 1$, $\beta = 1$ and $\rho = 0.5$), we now analyze the algorithms performance, focusing

Table 4.1 MMAS parameter configurations under study. Highlighted columns show the configuration parameters of the best overall configurations.

Config. Labels	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
ρ	0.5	0.5	0.5	0.5	0.5	0.5	0.1	0.1	0.1	0.1	0.1	0.1	0.02	0.02	0.02	0.02	0.02	0.02
α	1	1	1	2	2	2	1	1	1	2	2	2	1	1	1	2	2	2
β	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	1	2	3

on the adequacy of the encodings and influence of the pheromone and heuristic information.

In this section we analyze the performance of the two proposed algorithms with MTS heuristic information and different pheromone encodings, named as MMAS-NODE+H and MMAS-TIME+H. Besides, we study the power of the MTS heuristic by comparing the proposed approaches with their variants with the heuristic disabled (labelled as MMAS-NODE and MMAS-TIME) by setting the heuristic influence parameter $\beta = 0$. Analogously, we analyze the influence of MMAS pheromone learning mechanism by comparing the proposed versions with the variants with the pheromone influence disabled (labelled as *H*) by setting the pheromone influence parameter $\alpha = 0$, that is, solutions are only constructed using the heuristic information. Therefore, we analyze five variants whose parameters are specified in Table 4.2: the two proposed algorithms MMAS-NODE+H and MMAS-TIME+H, their variants without heuristic information MMAS-NODE and MMAS-TIME, and one variant that only uses heuristic information H. Note that there is only one variant with only heuristic information (H), as the encoding labels (NODE or TIME) show how the learned information is codified in the pheromone table.

Table 4.2 MMAS variants under analysis.

Short label	Pheromone table	Heuristic	α	β	ρ	M
MMAS-NODE+H	$\boldsymbol{\tau}_{\text{NODE}}$	✓	1	1	0.5	$8 \cdot N \cdot U$
MMAS-NODE	$\boldsymbol{\tau}_{\text{NODE}}$		1	0		
MMAS-TIME+H	$\boldsymbol{\tau}_{\text{TIME}}$	✓	1	1		
MMAS-TIME	$\boldsymbol{\tau}_{\text{TIME}}$		1	0		
H	-	✓	0	1	-	

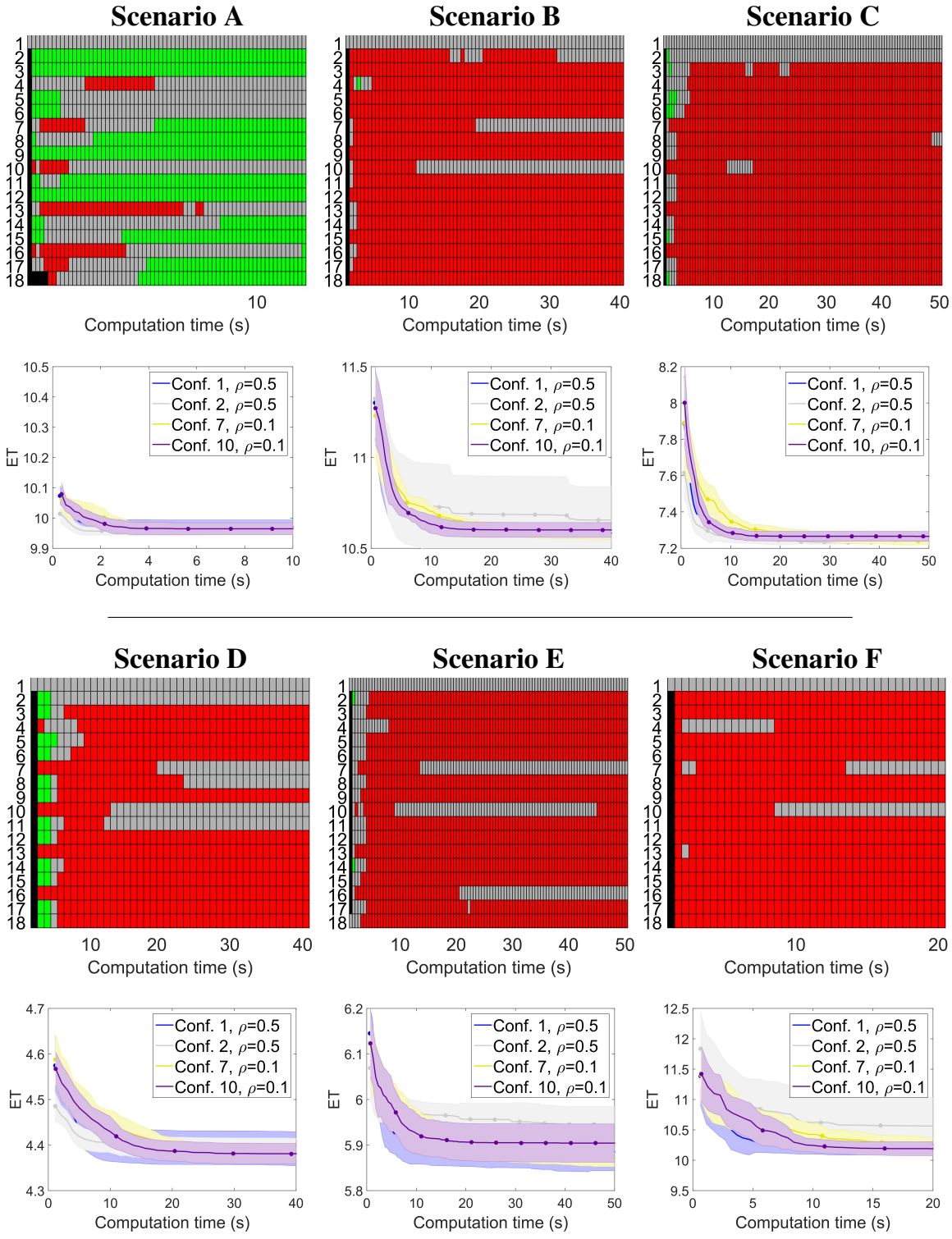


Figure 4.7 Comparison of all parameter configurations for MMAS-NODE+H. Configuration 1 is selected as base variant.

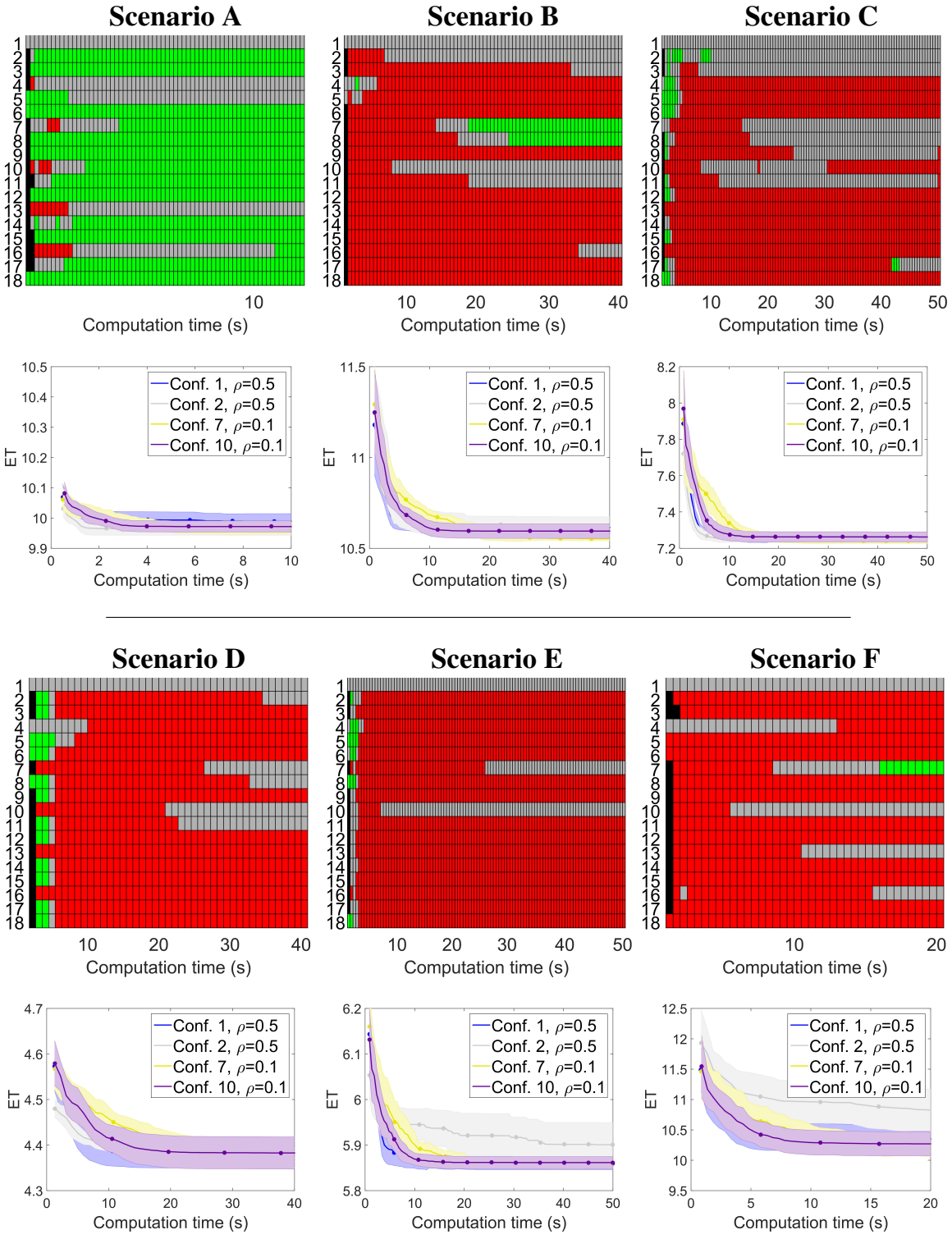


Figure 4.8 Comparison of all parameter configurations for MMAS-TIME+H. Configuration 1 is selected as base variant in the dominance evolution graphs.

The performance of all the variants over the six search scenarios are presented in Figure 4.9. For each scenario we represent the dominance evolution graphs of all the variants in the first row, and in the second and third rows respectively the ET evolution graphs for the variants with pheromones and with heuristic information.

On the one hand, the dominance evolution graphs show that the algorithm selected as the base (MMAS-NODE+H) dominates most of the considered variants. In particular, MMAS-NODE+H dominates the only-heuristic variant (H) and the two variants without heuristic (MMAS-NODE and MMAS-TIME) in all the scenarios, and shows a similar overall performance (depending on the scenario better, equal or worse) to MMAS-TIME+H.

On the other hand, the ET evolution graphs show the magnitude of the differences between the fitness values reached by all the variants along the iterations. These graphs show that the proposed variants (MMAS-NODE+H and MMAS-TIME+H) reach faster higher quality results than their variants without heuristic (MMAS-NODE and MMAS-TIME). The heuristic allows the proposed algorithms to converge faster in Scenario C and to find higher quality solutions from the first iterations in the rest of scenarios. Hence, we can conclude that the proposed MTS heuristic is appropriate for MTS and helps to reduce the computational times and find high quality solutions.

Moreover, from the comparison of the proposed algorithms with H variant without pheromone learning mechanism (displayed in the third graphic of each scenario) we can conclude that although the heuristic by itself is able to find high quality solutions quickly, the pheromone learning mechanism of MMAS enables to reach higher quality solutions faster in all scenarios.

Finally, regarding the two encodings analyzed, when the heuristic is disabled MMAS-NODE outperforms MMAS-TIME in Scenarios B and C. However, enabling the MTS heuristic helps MMAS-TIME+H to overcome the drawbacks of its encoding and when the heuristic is enabled the performances of both encodings are similar.

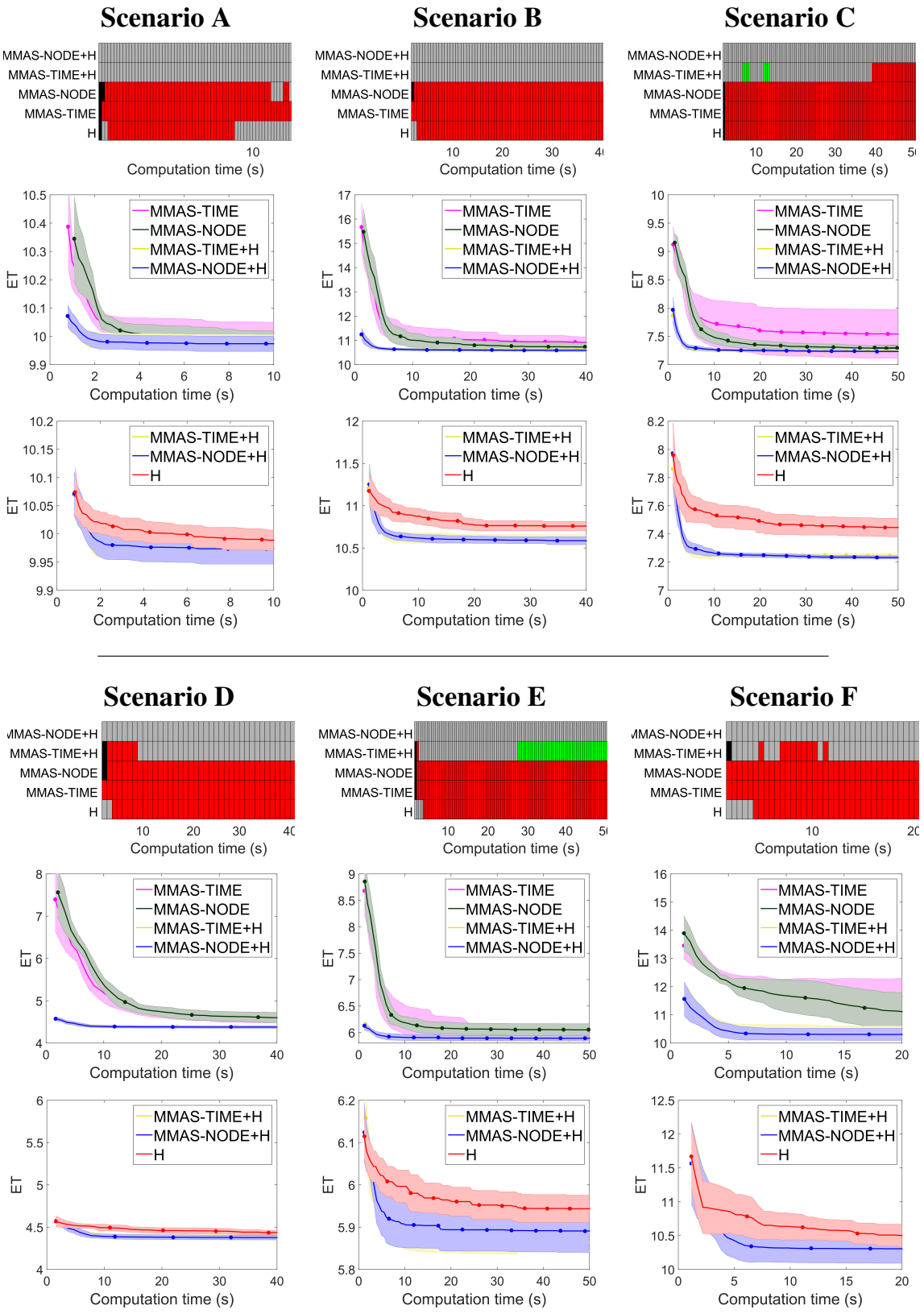


Figure 4.9 Comparison of MMAS variants with and without heuristic. The selected base variant in the dominance evolution graphs is MMAS-NODE+H.

4.3.3.3. Representative Solutions of our Approach

Figure 4.10 shows a representative solution obtained by MMAS-NODE+H for each search scenario under analysis. We have chosen the node encoding as the analysis in the previous section shows that it presents slightly better results than the time encoding.

Each graphic of Figure 4.10 contains the search trajectories represented with colored lines and sensor measurements with vertical yellow lines from the UAV position to the ground. Moreover, each solution displays the final "unnormalized belief" $\tilde{b}(v^N)$ represented with a colored matrix (where warmer colors indicate higher probabilities of target presence) and the fitness criterion (ET) and probability of detection (P_d) corresponding to each solution. It is worth noting that in most of the scenarios the UAVs manage to gather most of the probability. For instance in the solution of Scenario D, the three UAVs gathered more than 90 per cent of the probability ($P_d(s_{1:U}^{1:N}) = 0.928$). Besides, and due to this high gathered probability, the "unnormalized belief" $\tilde{b}(v^N)$ of many of the cells is significantly lower than their previous values in $b(v^0)$. Therefore the colormaps of Figure 4.10 have been rescaled respect the one used for representing the initial probability maps in Figure 4.6. For instance, in Scenario D the value of the cell in red with the highest $b(v^0)$ is 0.68 (see bottom left of Figure 4.6), while the corresponding value of the red colored cell with highest $\tilde{b}(v^N)$ is 0.019 (see bottom left of Figure 4.10).

The results of Figure 4.10 show that the search trajectories returned by the proposed MMAS-based algorithm overfly sooner the areas with higher chances of target presence (those with higher $\tilde{b}(v^t)$), making the UAVs to cooperate for gathering the "unnormalized belief" as soon as possible. For instance, in Scenario A the proposed MTS heuristic of MMAS-NODE+H makes the algorithm prefer the actions that lead the UAVs toward the slightly higher probability area on the south. Besides, in the dynamic Scenarios B and E, where the spreading probability movements complicate the search, the UAVs fly first to the areas where $\tilde{b}(v^t)$ is more concentrated and then follow the target movements. Moreover, in Scenario C both UAVs are distributed

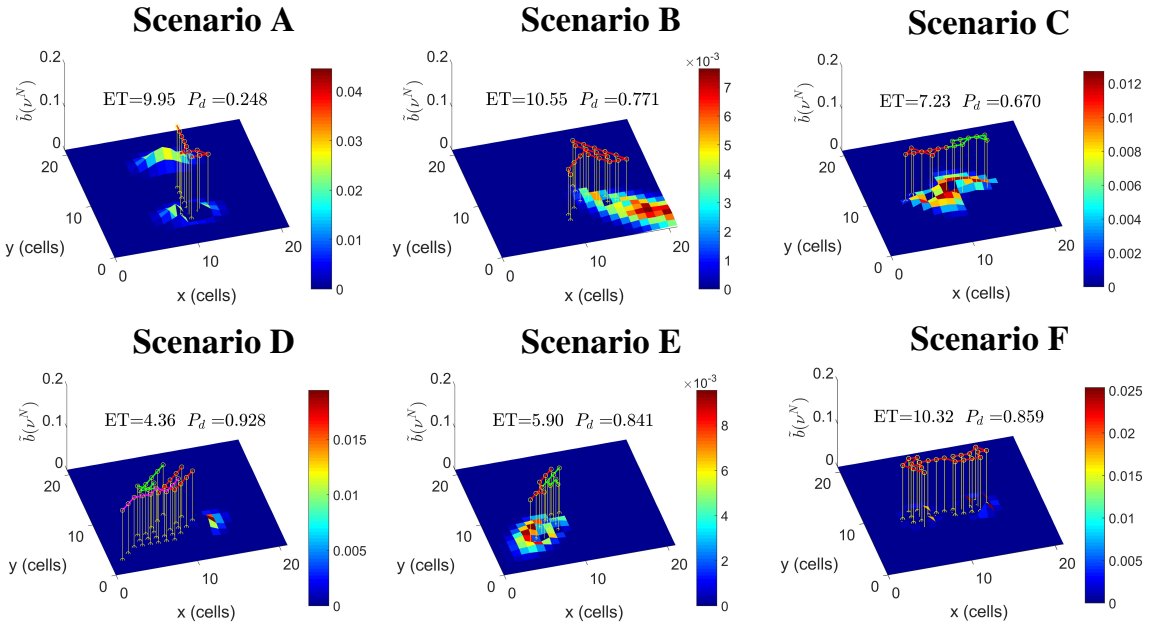


Figure 4.10 Representative solutions obtained with MMAS-NODE+H.

for searching the target, each one following one of the highest spreading probability areas. And in Scenario D, the three UAVs that carry out the search follow the target movements and manage to gather 92% of the initial probability. Finally, in Scenario F, the heuristic helps the algorithm to identify and flight first toward the higher probability area on the east and then make the UAV turn in the opposite direction toward the remaining high probability area on the west, finally gathering 86% of the initial probability.

4.3.4. Comparison with other MTS Approaches

In this section we compare the proposed MTS algorithm based on MMAS with other MTS approaches of the state of the art. Besides, we have selected the node encoding for the analysis as it produces slightly better results than the time encoding.

First, we compare our approach against the deterministic heuristic strategies proposed in (Meghjani et al., 2016) for the search for static targets by a single UAV. Then, we analyze the performance of three MTS algorithms based on different op-

timization approaches previously used in the literature (Lanillos et al., 2012, 2013; Lin and Goodrich, 2009).

4.3.4.1. Comparison with Ad-hoc MTS Heuristics

In this section we compare the stochastic MTS heuristic (H) that we propose in Section 4.2.2.2 and use in our MTS algorithms (MMAS-NODE+H and MMAS-TIME+H), against the three heuristics proposed by (Meghjani et al., 2016) for solving MTS problems where a single UAV searches for a static target.

4.3.4.1.1. Description of other MTS Heuristics. The three MTS heuristics under analysis are described below. It is worth noting that they have been proposed in (Meghjani et al., 2016) to deterministically obtain by themselves (without using an optimization technique) the UAV trajectory. Hence, their purpose is different to the MTS heuristic presented in this chapter, which was designed to be used as a constructive heuristic in MMAS. Besides, in contrast with the deterministic heuristic proposed in (Meghjani et al., 2016), due to the stochastic nature of our MTS heuristic it is convenient to run it several times.

Global Maximum Heuristic (HGM) generates a trajectory that tries to visit sequentially the cells with maximum probability of target presence by selecting at each time step the cardinal action that moves the UAVs toward the cell with the highest $\tilde{b}(v^t)$. Although this strategy may work well for beliefs that have the probability concentrated in one area, it is not adequate for beliefs whose probability is spread over several regions, as it makes the UAVs to move constantly from one high probability area to the others without collecting the remaining probability within each region. Besides, the heuristic can be directly employed for searching a dynamic target with multiple UAVs. However, in the case of multiple searchers this heuristic makes all the fleet to move toward the same cell.

Local Maximum Heuristic (HLM) directs the UAV toward the cell with maximum probability within a given maximum search radius r_{HLM}. It is worth noting that the

performance of this heuristic strongly depends on the initial UAV locations and is prone to get stuck into local minimum. Hence, in order to avoid this drawback the heuristic maximum search radius is increased $r_{HLM} \leftarrow r_{HLM} + \Delta_{HLM}$ when the cell with maximum $\tilde{b}(v^t)$ falls outside the UAV reach (defined by r_{HLM}) during several consecutive time steps N_{HLM}. Besides, this strategy can also be directly employed for searching a dynamic target with multiple UAVs. Moreover, the consideration of a maximum search radius can benefit the search making the UAVs to distribute in different regions.

Spiral Heuristic (HS) directs the UAV directly toward the cell with maximum probability and once there starts describing a spiral around the cell during N_{HS} time steps. Then, the UAVs is directed toward the cell with maximum $\tilde{b}(v^t)$ and again, once the cell is reached, the UAV starts describing a spiral during N_{HS} time steps. This process is repeated until the maximum number of time steps N is reached. It is worth noting that this strategy can not be applied straightforward to dynamic targets for big values of N_{HS}, as if a UAV describes a long spiral trajectory during multiple time steps the high probability region may have already move far away from the spiral covering region. In this case, the strategy could be modified considering a spiral whose center should be moved following the movements of the targets. However, for limited values of N_{HS} the strategy proposed by (Meghjani et al., 2016) can obtain good results.

4.3.4.1.2. Comparative Results with other Heuristics. In order to improve the heuristics explanation and to analyze their results, Figure 4.11 displays the solutions obtained by them for the three of the scenarios under analysis that show better their benefits and drawbacks. More concretely, Figure 4.11 contains the solutions obtained with HGM, HLM and HS in the three first rows and a representative solution of our proposed non-deterministic MTS heuristic (H) in the last row. The dominance and ET evolution graphs of Figure 4.12 compare the performance of the three previous heuristics, our heuristic (H) and the proposed optimization method (MMAS-NODE+H). Besides, the dominance evolution graphs have as base algorithm our pro-

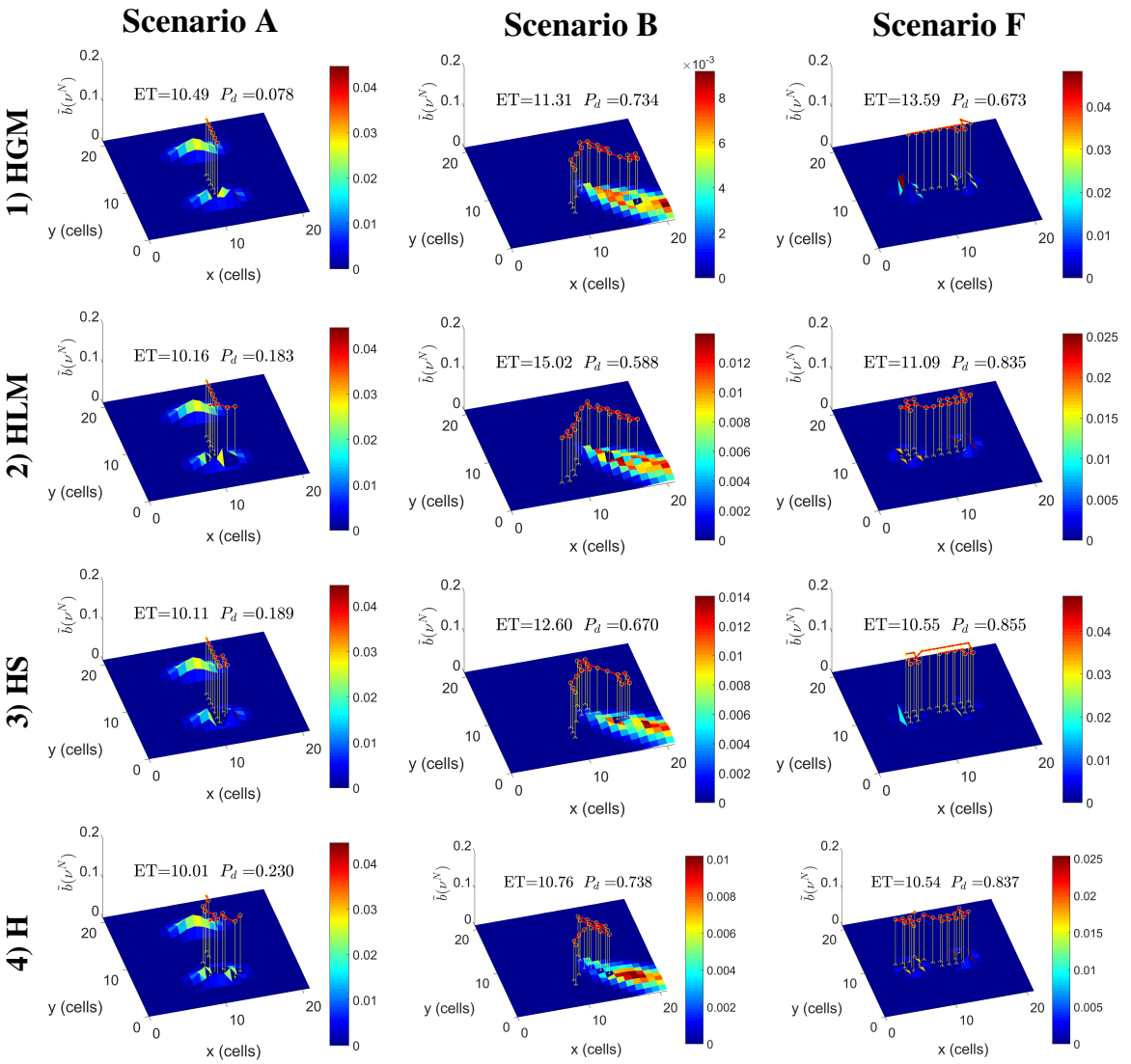


Figure 4.11 Solutions of the three MTS deterministic heuristics (HGM, HLM and HS) proposed in (Meghjani et al., 2016) and our proposed heuristic (H) for three of the analyzed scenarios.

posed MTS heuristic (H) and the ET evolution curves corresponding to the heuristic proposed in (Meghjani et al., 2016) are displayed with constant fitness along the computational time due to their deterministic nature. Furthermore, for all the results obtained in this section, we consider the following heuristics parameters: for HS a spiral length parameter of $N_{HS} = 5$ time steps, and for HLM a local reach radius $r_{HLM} = 5$, maximum consecutive time steps within a local minimum $N_{LHM} = 5$ and an increment of the reach radius of $\Delta_{HLM} = 5$.

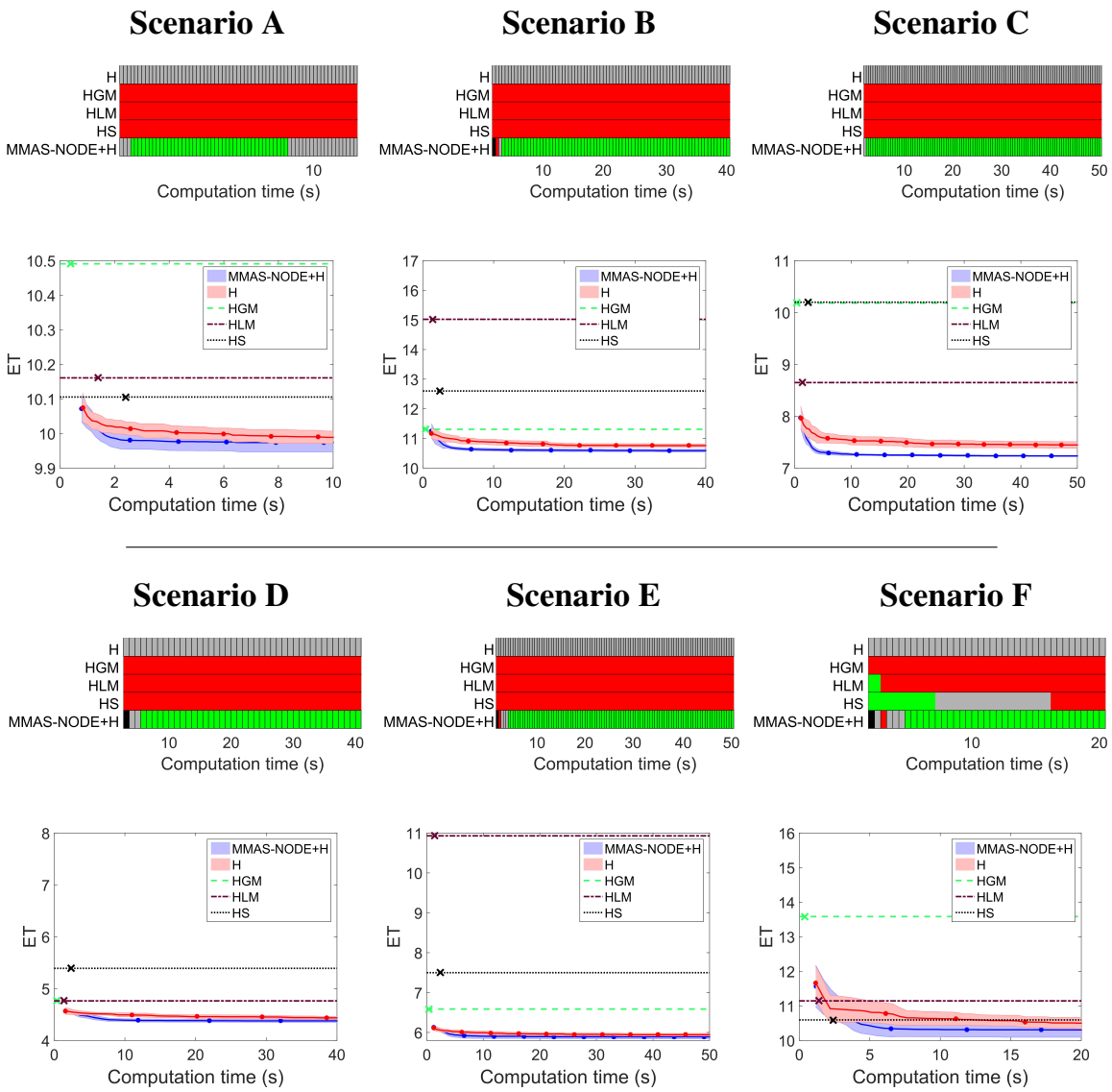


Figure 4.12 Comparison of the deterministic heuristics (HGM,HLM and HS) proposed in (Meghjani et al., 2016), our proposed MTS heuristic (H) and MMAS-NODE+H algorithm. The selected based variant in the dominance evolution graphs is H.

Regarding the global maximum heuristic (HGM), which sends the UAV toward the cell with maximum $\tilde{b}(v^t)$ at each time step, it happens to be an adequate strategy for the spreading movement of $\tilde{b}(v^t)$ of Scenario B, D and E, but as Figure 4.12 shows it presents low quality solutions for the other three scenarios. For instance, the trajectory obtained by HGM for Scenario A (displayed in the first column and first row of Figure 4.11) sends the UAV toward one of the maximum probability

cells and, afterwards, it makes the UAV return the same way toward the other high probability cell (placed in the other probability region), in spite of the fact that this second cell is not reachable within the limited horizon time N. Besides, the HGM trajectory for Scenario F (in first row and third column of Figure 4.11) initially sends the UAV toward the cell with maximum $b(v^0)$ situated in the higher probability area on the east, then toward the other high probability area on the west and finally again toward the one in the east, loosing a lot of time flying between high probability areas and obtaining the worst ET value among the one obtained for the different heuristics in this scenario (compare the ET values in the third column of Figure 4.11).

Respect the local maximum heuristic (HLM), it proposes good search trajectories for Scenario D and F, but obtains bad performance in Scenario B and E. As it can be seen in the solutions of the second row of Figure 4.11, the locality of this heuristic benefits the search in Scenario A, preventing the UAV from trying to reach the two high probability areas (in this scenario there is only time available for exploring one of the high probability areas). However, in Scenario B the locality has myopic effects during the first time steps of the search, where the cells with non null $\tilde{b}(v^t)$ are further from the UAV reached determined by r_{HLM}. Nevertheless, once $\tilde{b}(v^t)$ starts spreading following the movements given by $P(v^t|v^{t-1})$ some $\tilde{b}(v^t) \neq 0$ falls inside the UAV reach and the UAV is able to start following the target movements. Eventually, in Scenario F the UAV starts gathering probability of the higher probability area on the east, and then, when the maximum number of consecutive time steps within a local minimum is fulfilled, the UAV reach radius is increased by $r_{HLM} \leftarrow r_{HLM} + \Delta_{HLM}$ and the UAV is sent toward the other high probability area, obtaining lower ET than HGM.

With regard to the spiral trajectory (HS), it produces good search trajectories in Scenarios A and F. The solutions displayed in the third row of Figure 4.11 show that for Scenario A the UAV is initially directed toward the cell with maximum $\tilde{b}(v^t)$ and once this cell is reached, it performs a spiral trajectory of length $N_{HS} = 5$, obtaining a expected time (ET=10.11) lower than HGM and HLM. Besides, in Scenario F, the trajectory proposed by HS makes a spiral around the high probability areas

obtaining a low fitness value (ET=10.54). Lastly, in Scenario B the UAV is able to reach the current cell with highest probability twice and thus the UAV makes two spirals obtaining a medium quality trajectory (ET=12.60), which is better than the one obtained by HLM.

Regarding the methods proposed in this thesis (H and MMAS-NODE+H), on the one hand, Figure 4.12 shows that our MTS heuristic obtains after a few seconds better performance than the three heuristics proposed by (Meghjani et al., 2016) in all scenarios. As it can be seen in the solutions displayed in the fourth row of Figure 4.11, the heuristic sends the UAV toward the close high probability areas and is able to follow the target movements obtaining low ET values. On the other hand, Figure 4.12 also shows that the proposed algorithm MMAS-NODE+H presents even better results than all the considered heuristics in all the scenarios. This happens because, as the solutions returned by MMAS-NODE+H represented in Figure 4.10 show, the pheromones help the ants to learn from good previous solutions, finally obtaining better search trajectories.

To conclude, although the three heuristics proposed in (Meghjani et al., 2016) present a good performance in some scenarios, they obtain low quality solutions in others. Hence, the performance of HGM, HLM and HS strongly depends on the search scenario. On the contrary, our MTS heuristic obtains better performance than the three heuristics in all the scenarios, and when H is combined with MMAS pheromone mechanism, MMAS-NODE+H requires a few seconds of computational time to improve the performance of all the heuristics.

4.3.4.2. Comparison with other Optimization Methods

In this section we compare our MTS algorithm based on MMAS against three other approaches. Two of them are based on the only two optimization methods that, appearing in the literature reviewed in Chapter 2, have been previously used successfully for optimizing the expected target detection time in discrete domains. The third one has been selected because it is a well known approach, already used to optimize

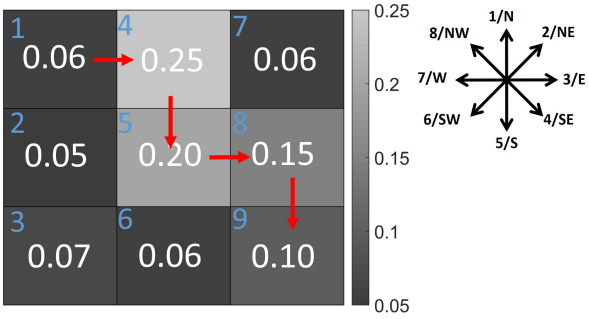


Figure 4.13 Simple search scenario with $w_x \times w_y = 3 \times 3$ cells, where blue numbers identify each cell and the white ones display the initial belief $b(v^0)$ within them. The search trajectory defined by initial cell $s_1^0 = 1$ and cardinal actions $c_1^{1:4} = \{3,5,3,5\}$ is represented with red arrows.

the probability of target detection, that can be set up in a straightforward way for optimizing the expected target detection time. First, we introduce the considered MTS algorithms based on Cross Entropy Optimization (CEO), Bayesian Optimization Algorithm (BOA) and Genetic Algorithm (GA). Then we analyze their similarities and differences with the ACO based approach followed in this thesis. And finally, we compare the performance of all the MTS algorithms over the considered search scenarios.

4.3.4.2.1. Description of the other MTS Algorithms (CEO, BOA and GA). The characteristics of the other three MTS discrete algorithms used in this thesis to compare our new approach against them are explained below from the general idea of the techniques to the details of their MTS implementation.

Besides, with the purpose of clarifying the MTS implementations of each approach, we use the simple search scenario represented in Figure 4.13 for the description of the algorithms. In this scenario the initial probability of target presence within each cell of a 3x3 rectangular grid is represented with white numbers over the cells and by a gray colormap. Besides, the optimal search trajectory $c_1^{1:4} = \{3,5,3,5\}$ for a unique UAV starting at $s_1^0 = 1$ and with a planning horizon of length N=4 is represented with red arrows.

Cross Entropy Optimization (CEO) is a technique that uses adaptive importance sampling for solving optimization problems (Rubinstein, 1999). This iterative algorithm attempts to learn the optimal solution distribution from the best solutions found at each algorithm iteration. New solutions are randomly sampled from the current estimation of the optimal solution distribution at iteration k, which is named $\hat{\mathbf{p}}^k_{CEO}$ and whose hat indicates that is an estimated probability distribution. Initially, the population of solutions are generated randomly and the probability distribution is initialized with a uniform distribution (initially it does not contain any knowledge about the problem solution). At each iteration of CEO, first the population of solutions is sampled from previous $\hat{\mathbf{p}}^k_{CEO}$ and then the probability distribution is reestimated using the solutions of the iteration with better fitness and assuming probability independence between the problem variables.

CEO was first applied to MTS in (Lanillos et al., 2012), where $\hat{\mathbf{p}}^k_{CEO}$ represents the probability that each UAV takes each of the 8 possible cardinal directions at each instant t within the sequence of N high-level commands. Therefore, when the search is carried out by a unique UAV, $\hat{\mathbf{p}}^k_{CEO}$ can be stored in a $8 \times N$ matrix, where each element represents the probability of taking the action of its row index at the time step of its column index. Besides, when solving search scenarios with multiple UAVs, $\hat{\mathbf{p}}^k_{CEO}$ can be saved in $8 \times N \times U$ matrix, where the actions of each UAV are stored in a different slice of the matrix.

The pseudocode of the CEO-based MTS planner is schematized in Algorithm 5. Apart from the MTS related inputs, CEO also requires the population size R, the percentage ε of better solutions used to estimate $\hat{\mathbf{p}}^k_{CEO}$, and a smoothing parameter γ to avoid abrupt changes of $\hat{\mathbf{p}}^k_{CEO}$. During the initial steps, $\hat{\mathbf{p}}^k_{CEO}$ is uniformly distributed (to make equiprobable any of the eight cardinal actions at all time steps), and the best found solution ${}^{gb}\mathbf{s}$, its value ${}^{gb}\mathbf{ET}$, the number of solutions required to estimate $\hat{\mathbf{p}}^k_{CEO}$ and the iteration index k are initialized. The main loop (from line 5 to line 17) is iterated until the maximum computational time is reached. At the beginning of each algorithm iteration a new population of R solutions (sequences of UAVs actions) is sampled according to $\hat{\mathbf{p}}^k_{CEO}$ (lines 7), their trajectory and ET obtained using the UAV

Algorithm 5 Cross Entropy Optimization (CEO)

Require: $N, s^0_{1:U}$ ▷ Number of control actions and initial UAVs locations
Require: $P(\nu^0)$, $P(\nu^t|\nu^{t-1})$, $P(\overline{D}^t|\nu^t, s^t_u), s^{t+1}_u = f(s^t_u, c^t_u)$ ▷ Target and UAV models
Require: R, ε, γ, ▷ CEO parameters: number of samples, percentage and smoothing
1: ${}^{gb}\boldsymbol{ET} \leftarrow \infty$, ${}^{gb}\boldsymbol{s} \leftarrow []$ ▷ Set initial best objective value and solution
2: $\hat{\mathbf{p}}^k_{CEO} \leftarrow$ Initialize uniformly the probability distribution ▷ Initialize the probability distribution
3: $E \leftarrow R \cdot \varepsilon$ ▷ Number of solutions used to estimate $\hat{\mathbf{p}}^k_{CEO}$
4: $k \leftarrow 0$ ▷ Set iteration index
5: **while** no finished **do**
6: **for** r=1:R **do**
7: $c^{1:N}_{1:U} \sim \hat{\mathbf{p}}^k_{CEO}$ ▷ Sample concatenated sequences of high level commands
8: $s^{0:N}_{1:U} \leftarrow ObtainTrajectories(s^0_{1:U}, c^{1:N}_{1:U}, s^{t+1}_u = f(s^t_u, c^t_u))$ ▷ Obtain UAV trajectories
9: $ET(s^{0:N}_{1:U}) \leftarrow EvaluateET(s^{0:N}_{1:U}, P(\nu^0), P(\nu^t|\nu^{t-1}), P(\overline{D}^t|\nu^t, s^t_u))$ ▷ Evaluate
10: $\boldsymbol{s}_r \leftarrow s^{0:N}_{1:U}$, $\boldsymbol{c}_r \leftarrow c^{1:N}_{1:U}$, $\boldsymbol{ET}_r \leftarrow ET(s^{0:N}_{1:U})$ ▷ Store current information
11: **end for**
12: $[{}^{gb}\boldsymbol{ET}, {}^{gb}\boldsymbol{s}] \leftarrow SelectBest([{}^{gb}\boldsymbol{ET}, \boldsymbol{ET}_{1:R}], [{}^{gb}\boldsymbol{s}, \boldsymbol{s}_{1:R}])$ ▷ Select global best solution
13: $[\boldsymbol{ET}_{1:E}, \boldsymbol{c}_{1:E}] \leftarrow SelectBetter(\boldsymbol{ET}_{1:R}, \boldsymbol{c}_{1:R}, E)$ ▷ Select best subset of E solutions
14: $\hat{\mathbf{p}}^{k+1}_{CEO} \leftarrow LearnDistributionCounting(\boldsymbol{c}_{1:E})$ ▷ Count command types in each time step
15: $\hat{\mathbf{p}}^{k+1}_{CEO} \leftarrow \gamma\, \hat{\mathbf{p}}^{k+1}_{CEO} + (1-\gamma)\hat{\mathbf{p}}^k_{CEO}$ ▷ Smooth the probability distribution
16: $k \leftarrow k+1$ ▷ Update the iteration index
17: **end while**
18: **return** ${}^{gb}\boldsymbol{ET}$, ${}^{gb}\boldsymbol{s}$ (solution with minimum ET)

and target models (lines 8 and 9), and their values stored (line 10). Next, if a solution with better fitness than the previous global best solution is found, ${}^{gb}\boldsymbol{ET}$ and ${}^{gb}\boldsymbol{s}$ are updated (line 12). Afterwards, in line 13 a percentage ε of the sequences of actions with better ET are selected and used in line 14 to learn a new $\hat{\mathbf{p}}^{k+1}_{CEO}$ by counting how often each of the 8 commands occurs in each time step for each UAV. Besides, $\hat{\mathbf{p}}^{k+1}_{CEO}$ is smoothed in line 15, according to the smoothing parameter γ, towards its previous iteration value to avoid abrupt changes in the probability distribution. Finally, once the stop condition is reached, the algorithm returns the best found solution ${}^{gb}\boldsymbol{s}$ with fitness ${}^{gb}\boldsymbol{ET}$.

Figure 4.14 displays an illustrative example of how CEO estimates $\hat{\mathbf{p}}^k_{CEO}$ for the simple scenario of Figure 4.13. As in this scenario only one UAV carries out the search and the time horizon is $N = 4$, the distribution $\hat{\mathbf{p}}^k_{CEO}$ is saved in a 8×4 matrix.

Figure 4.14 displays the initial uniform distribution $\hat{\mathbf{p}}^0_{CEO}$ and the reestimated distribution after the fourth iteration of the algorithm $\hat{\mathbf{p}}^4_{CEO}$. While the initial distribution assigns a probability of 1/8 to all of the cardinal actions, after the fourth algorithm iteration it can observed how $\hat{\mathbf{p}}^4_{CEO}$ has already learned the optimal trajectory assigning higher probabilities to the actions corresponding to the optimal action sequence $c_1^{1:4} = \{3,5,3,5\}$.

$$\hat{\mathbf{p}}^0_{CEO} = \begin{array}{c} \text{time steps} \\ \begin{bmatrix} 0.1250 & 0.1250 & 0.1250 & 0.1250 \\ 0.1250 & 0.1250 & 0.1250 & 0.1250 \\ 0.1250 & 0.1250 & 0.1250 & 0.1250 \\ 0.1250 & 0.1250 & 0.1250 & 0.1250 \\ 0.1250 & 0.1250 & 0.1250 & 0.1250 \\ 0.1250 & 0.1250 & 0.1250 & 0.1250 \\ 0.1250 & 0.1250 & 0.1250 & 0.1250 \\ 0.1250 & 0.1250 & 0.1250 & 0.1250 \end{bmatrix} \end{array} \text{actions} \qquad \hat{\mathbf{p}}^4_{CEO} = \begin{array}{c} \text{time steps} \\ \begin{bmatrix} 0.0032 & 0.0118 & 0.0115 & 0.2380 \\ 0.0032 & 0.0032 & 0.0038 & 0.0392 \\ \boxed{0.9637} & 0.0132 & \boxed{0.6229} & 0.0129 \\ 0.0171 & 0.1238 & 0.1600 & 0.0291 \\ 0.0032 & \boxed{0.8366} & 0.0432 & \boxed{0.2326} \\ 0.0032 & 0.0032 & 0.0233 & 0.1607 \\ 0.0032 & 0.0038 & 0.1205 & 0.1847 \\ 0.0032 & 0.0044 & 0.0147 & 0.1027 \end{bmatrix} \end{array} \text{actions}$$

Figure 4.14 Example of evolution of $\hat{\mathbf{p}}^k_{CEO}$ in a MTS with the 3x3 belief map of Figure 4.13, a single UAV and a decision horizon of $N = 4$ values. Its optimal trajectory, known due to the simplicity of the problem, is $c_1^{1:4} = \{3,5,3,5\}$.

Bayesian Optimization Algorithm (BOA) also estimates the distribution of promising solutions, learning the Bayesian Networks (BN) that better describes the current distribution of the solutions (Pelikan et al., 1999). A BN is a probabilistic graphical model whose structure and probability tables encode the relationships between the variables (solution components). Therefore, BOA does not assume the independence between the variables (presupposed, on the other hand, by CEO). Besides, BOA permits to initialize the BN structure with a fully disconnected graph or considering prior information about the relationships between the variables.

The MTS algorithm based on BOA first used in (Lanillos et al., 2013) follows a similar strategy than the CEO-based algorithm: it learns the probability distribution of the best actions to take at each time step. Similarly, at each iteration of BOA, the solutions are sampled from $\hat{\mathbf{p}}_{BOA}$ (that follows a uniform distribution at the first iteration) and then $\hat{\mathbf{p}}_{BOA}$ is re-estimated using the information contained in the best

solutions found by the algorithm. The graph model of the BN entails the dependencies among the variables (UAV control actions) and its parameters are the probability tables associated to each variable given their parents in the BN graph. Therefore, in order optimize MTS, BOA needs to solve a second optimization problem: finding the BN that fits better the data associated to the best solutions found by BOA, where "fits better" is defined by a metric that states how well the data (solutions in BOA) fit the BN. In this thesis, similarly to (Lanillos et al., 2013) and (Pelikan et al., 1999), we use the Bayesian Dirichlet metric (Neapolitan et al., 2004) .

The pseudocode of the BOA-based MTS planner is schematized in Algorithm 6. The algorithm starts by initializing the global best information (${}^{gb}\boldsymbol{s}$ and ${}^{gb}\boldsymbol{ET}$), the probability distribution ($\hat{\boldsymbol{p}}^0_{BOA}$), the number of solutions (E) in the best subset defined by the percentage ε as well as their control sequences and fitness values (${}^{s}\boldsymbol{c}_{1:E}$ and ${}^{s}\boldsymbol{ET}_{1:E}$), and the iteration index k. Within the main loop (lines 6 to 17), first the population of sequences of actions is sampled (line 8), their associated trajectories and ET obtained (lines 9 and 10), and their values stored (line 11). Besides, to avoid abrupt changes within the probability distribution obtained in two consecutive iterations of BOA, $\hat{\mathbf{p}}^k_{BOA}$ is estimated (in line 15) from the best solutions ${}^{s}\boldsymbol{c}_{1:E}$ found during all iterations, which are selected (in line 14) among the new population of solutions and the previous selected ones. Once the maximum computational time is reached, BOA returns the best global best trajectory ${}^{gb}\mathbf{s}$ and its fitness ${}^{gb}\mathbf{ET}$.

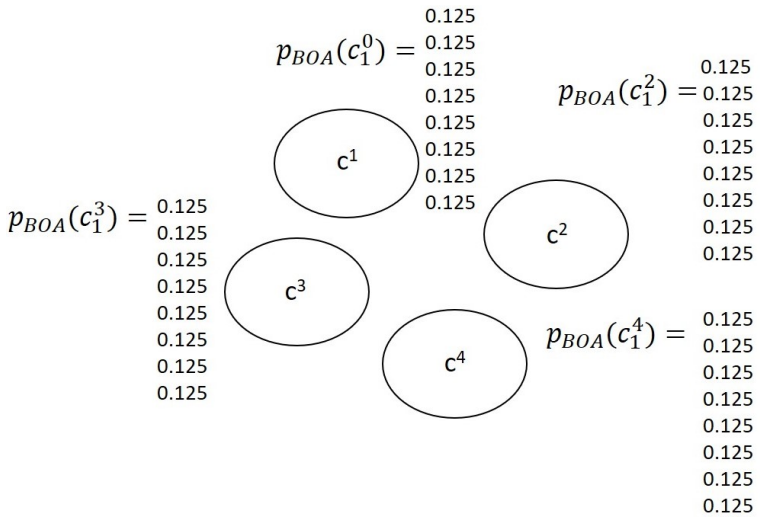
Finally, it is worth noting that one of the bottlenecks of BOA is the estimation of the BN. In particular, learning the structure of the BN is a NP-hard optimization problem whose complexity grows with the number of decision variables in the problem ($N \cdot U$ in our case). To reduce the computational cost of this learning process, it is possible to impose certain restrictions on the possible BN structures. In our case, we can exploit the fact that the future actions c_u^{t+1} of the control sequence are somehow dependent (through the previous "unnormalized belief" $\tilde{b}(v^t)$) on the past actions $c_u^{0:t}$. For this reason, the BN learning approach used in this thesis within BOA is K2, a method proposed in (Cooper and Herskovits, 1992) that assumes an ancestral

Algorithm 6 Bayesian Optimization Algorithm (BOA)

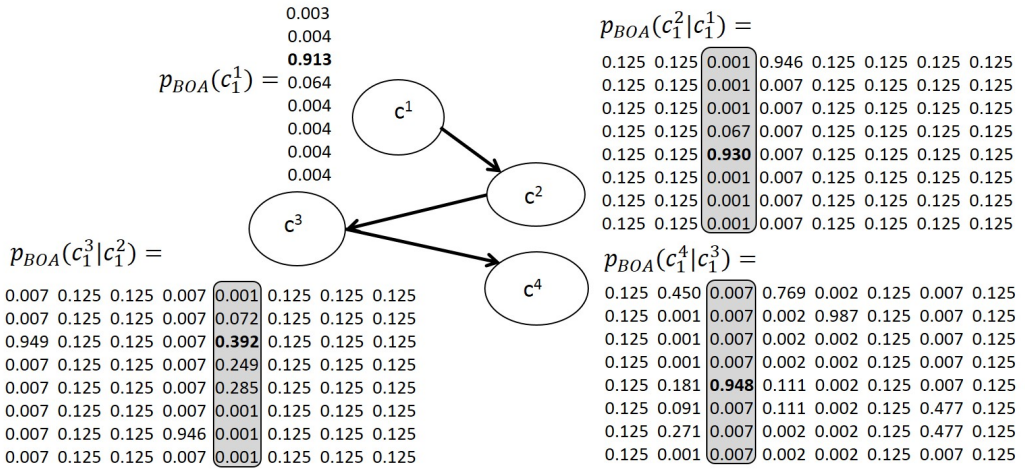
Require: $N, s^0_{1:U}$ ▷ Number of control actions and initial UAVs locations
Require: $P(\mathbf{v}^0)$, $P(\mathbf{v}^t|\mathbf{v}^{t-1})$, $P(\overline{D}^t|\mathbf{v}^t, s^t_u)$, $s^{t+1}_u = f(s^t_u, c^t_u)$ ▷ Target and UAV models
Require: R, ε ▷ BOA parameters: number of samples and percentage
$^{gb}\boldsymbol{ET} \leftarrow \infty$, $^{gb}\boldsymbol{s} \leftarrow []$ ▷ Set initial best objective value and solution
$\hat{\mathbf{p}}^k_{BOA} \leftarrow$ Initialize the probability distribution ▷ Initialize the probability distribution
$E \leftarrow R \cdot \varepsilon$ ▷ Number of solutions to estimate $\hat{\mathbf{p}}^k_{CEO}$ from
$^{s}\boldsymbol{c}_{1:E} \leftarrow \emptyset$, $^{s}\boldsymbol{ET}_{1:E} \leftarrow \emptyset$ ▷ Initialize set of selected control actions and objective values
$k \leftarrow 0$ ▷ Set iteration index
while no finished **do**
  for r=1:R **do**
    $c^{0:N}_{1:U} \sim \hat{\mathbf{p}}^k_{BOA}$ ▷ Sample concatenated sequences of high level commands
    $s^{0:N}_{1:U} \leftarrow ObtainTrajectories(s^0_{1:U}, c^{1:N}_{1:U}, s^{t+1}_u = f(s^t_u, c^t_u))$ ▷ Obtain UAV trajectories
    $ET(s^{0:N}_{1:U}) \leftarrow EvaluateET(s^{0:N}_{1:U}, P(\mathbf{v}^0), P(\mathbf{v}^t|\mathbf{v}^{t-1}), P(\overline{D}^t|\mathbf{v}^t, s^t_u))$ ▷ Evaluate
    $\boldsymbol{s}_r \leftarrow s^{0:N}_{1:U}$, $\boldsymbol{c}_r \leftarrow c^{1:N}_{1:U}$, $\boldsymbol{ET}_r \leftarrow ET(s^{0:N}_{1:U})$ ▷ Store current information
  end for
  $[^{gb}\boldsymbol{ET}, ^{gb}\boldsymbol{s}] \leftarrow SelectBest([^{gb}\boldsymbol{ET}, \boldsymbol{ET}_{1:R}], [^{gb}\boldsymbol{s}, \boldsymbol{s}_{1:R}],)$ ▷ Select global best solution
  $[^{s}\boldsymbol{ET}_{1:E}, ^{s}\boldsymbol{c}_{1:E}] \leftarrow SelectBetter([\boldsymbol{ET}_{1:R}, ^{s}\boldsymbol{ET}_{1:E}], [\boldsymbol{c}_{1:R}, ^{s}\boldsymbol{c}_{1:E}], E)$ ▷ Select best subset
  $\hat{\mathbf{p}}^k_{BOA} \leftarrow LearnBN(^{s}\boldsymbol{c}_{1:E})$ ▷ Learn the Bayesian network
  $k \leftarrow k+1$ ▷ Update the iteration index
end while
return $^{gb}\boldsymbol{ET}, ^{gb}\boldsymbol{s}$ (solution with minimum ET)

ordering among the BN variables. Hence, the selected approach is quicker than the generic BN learning strategy applied in (Lanillos et al., 2013).

In order to illustrate how BOA estimates $\hat{\mathbf{p}}^k_{BOA}$, we use again the simple scenario of Figure 4.13. As this scenario only has one UAV and a planning horizon of $N = 4$, $\hat{\mathbf{p}}^k_{BOA}$ is a BN with 4 discrete decision variables that can take values between 1 to 8. Initially, and as Figure 4.15 (a) shows, the nodes (variables) of the BN are unconnected and the probabilities tables of each variable are assigned a equiprobable value of 1/8, making all cardinal commands (each represented in one row) at each time step independent of each other and equally probable. Figure 4.15 (b) displays the probability distribution $\hat{\mathbf{p}}^8_{BOA}$ after only eight iterations of the algorithm. In particular, the arrows in the BN graph show the relationships learned between the variables (each command depends on the previous) and the two dimensional probability tables



(a) $\hat{\mathbf{p}}^0_{BOA}$



(b) $\hat{\mathbf{p}}^8_{BOA}$

Figure 4.15 Estimation of probability distributions $\hat{\mathbf{p}}^k_{BOA}$ and BN graph structure for a MTS with the 3x3 belief map of Figure 4.13. (a) $\hat{\mathbf{p}}^k_{BOA}$ at the initial step of the algorithm. (b) $\hat{\mathbf{p}}^k_{BOA}$ after eight iteration of the algorithm. Highlighted elements in bold correspond to the optimal trajectory $c_1^{1:4} = \{3,5,3,5\}$.

display how the probability of a given command changes (along the rows of the table) according to the values of the previous step command (each value represented in a different column). If we analyze the cardinal actions with higher probability at each time step t given the best command at the previous time step $t-1$, we observe

that they correspond to the high level command sequence associated to the optimal trajectory $c_1^{1:4} = \{3,5,3,5\}$.

Genetic Algorithm (GA) is a widely known iterative algorithm inspired in the process of natural evolution proposed by (Holland, 1992). In GA the population of solutions improves iteration by iteration by means of selection, crossover and mutation operators inspired in the evolution process of the genes. Besides, in contrast to the other techniques under analysis, this technique does not usually learn a probability distribution.

GA has been previously employed in (Lin and Goodrich, 2009) for maximizing the probability of detection (P_d) of a static target and in (Pérez-Carabaza et al., 2018) for minimizing the ET of dynamic or static targets. (Lin and Goodrich, 2009) proposes two codification of solutions; either by a sequence of nodes or by a sequence of cardinal actions. The node codification has two drawbacks: the resulting trajectories after the crossover may have different lengths and the parents must have at least a common cell to be able to cross them. For this reason, as both codifications showed similar performance, we select the latter codification, that is also similar to the one used in other approaches. Furthermore, we consider a binary-tournament selection operator to choose the parents of the new population, a single point crossover operator for combining the solutions of two parents, and a uniform mutation towards one of the two cardinal actions surrounding the existing one in the solution.

The pseudocode of the GA-based MTS planner is schematized in Algorithm 7. Apart from the MTS inputs the algorithm requires values for the population size R and the probability that a pair of parents of the population is crossed p_{xover} or a gene (decision variable) mutated p_{mut}. Besides, we set the number of children of each iteration equal to the number of individuals in each generation. In the initial steps, the best global information (${}^{gb}\boldsymbol{ET}$ and ${}^{gb}\boldsymbol{s}$) is initialized (line 1), the initial population is randomly generated (line 2), and their associated trajectories and fitness values evaluated and stored (lines 3 to 7). Within the main loop (lines 9 to 21), first the parents are selected according to the binary-tournament selection operator, which chooses each parent as the one with better fitness among two randomly selected

Algorithm 7 Genetic Algorithm (GA)

Require: $N, s^0_{1:U}$ ▷ Number of control actions and initial UAVs locations
Require: $P(\nu^0)$, $P(\nu^t|\nu^{t-1})$, $P(\overline{D}^t|\nu^t, s^t_u)$, $s^{t+1}_u = f(s^t_u, c^t_u)$ ▷ Target and UAV models
Require: R, p_{xover}, p_{mut} ▷ GA parameters: population size and crossover and mutation probabilities
1: ${}^{gb}\boldsymbol{ET} \leftarrow \infty$, ${}^{gb}\boldsymbol{s} \leftarrow []$ ▷ Set initial best objective value and solution
2: $\boldsymbol{c}_{1:R} \leftarrow$ Initialize randomly values between 1:8 ▷ Initialize population of solutions
3: **for** r=1:R **do**
4: $s^{0:N}_{1:U} \leftarrow ObtainTrajectories(s^0_{1:U}, \boldsymbol{c}_r, s^{t+1}_u = f(s^t_u, c^t_u))$ ▷ Obtain UAV trajectories
5: $ET(s^{0:N}_{1:U}) \leftarrow EvaluateET(s^{0:N}_{1:U}, P(\nu^0), P(\nu^t|\nu^{t-1}), P(\overline{D}^t|\nu^t, s^t_u))$ ▷ Evaluate
6: $\boldsymbol{s}_r \leftarrow s^{0:N}_{1:U}$, $\boldsymbol{ET}_r \leftarrow ET(s^{0:N}_{1:U})$ ▷ Store current information
7: **end for**
8: $k \leftarrow 0$ ▷ Set iteration index
9: **while** no finished **do**
10: $[\boldsymbol{c}^{parents1}_{1:R/2}, \boldsymbol{c}^{parents2}_{1:R/2}] \leftarrow SelectParents(\boldsymbol{c}_{1:R}, \boldsymbol{ET}_{1:R})$ ▷ Select pairs of parents
11: $\boldsymbol{c}^{children}_{1:R} \leftarrow Crossover(\boldsymbol{c}^{parents1}_{1:R/2}, \boldsymbol{c}^{parents2}_{1:R/2}, p_{xover})$ ▷ Cross parents to obtain children
12: $\boldsymbol{c}^{children}_{1:R} \leftarrow Mutate(\boldsymbol{c}^{children}_{1:R}, p_{mut})$ ▷ Mutate children
13: **for** r=1:R **do**
14: $s^{0:N}_{1:U} \leftarrow ObtainTrajectories(s^0_{1:U}, \boldsymbol{c}^{children}_r, s^{t+1}_u = f(s^t_u, c^t_u))$ ▷ Obtain trajectories
15: $ET(s^{0:N}_{1:U}) \leftarrow EvaluateET(s^{0:N}_{1:U}, P(\nu^0), P(\nu^t|\nu^{t-1}), P(\overline{D}^t|\nu^t, s^t_u))$ ▷ Evaluate
16: $\boldsymbol{s}^{children}_r \leftarrow s^{0:N}_{1:U}$, $\boldsymbol{ET}^{children}_r \leftarrow ET(s^{0:N}_{1:U})$ ▷ Store current information
17: **end for**
18: $[{}^{gb}\boldsymbol{ET}, {}^{gb}\boldsymbol{s}] \leftarrow SelectBest([{}^{gb}\boldsymbol{ET}, \boldsymbol{ET}^{children}_{1:R}], [{}^{gb}\boldsymbol{s}, \boldsymbol{s}^{children}_{1:R}])$
19: $[\boldsymbol{ET}_{1:R}, \boldsymbol{c}_{1:R}] \leftarrow Survivors([\boldsymbol{ET}_{1:R}, \boldsymbol{ET}^{children}_{1:R}], [\boldsymbol{c}_{1:R}, \boldsymbol{c}^{children}_{1:R}])$ ▷ Select survivors
20: $k \leftarrow k+1$ ▷ Update the iteration index
21: **end while**
22: **return** ${}^{gb}\boldsymbol{ET}$, ${}^{gb}\boldsymbol{s}$ (solution with minimum ET)

members of the population (line 10). Next, a portion of the children, determined by p_{xcross}, are mixed by the single point crossover operator and the values of some of the children genes (solution components) are changed by a uniform mutation operator according to the probability p_{mut}. Next the offspring population is simulated and evaluated (in lines 13 to 17). Then the survivors for the next population are selected according to an elitist survivor operator that selects the best R solutions/ants among the old $\boldsymbol{c}_{1:R}$ and new populations $\boldsymbol{c}^{children}_{1:R}$. This process is repeated until the stop condition is reached and the algorithm returns the best found solution ${}^{gb}\mathbf{s}$ and its fitness ${}^{gb}\mathbf{ET}$.

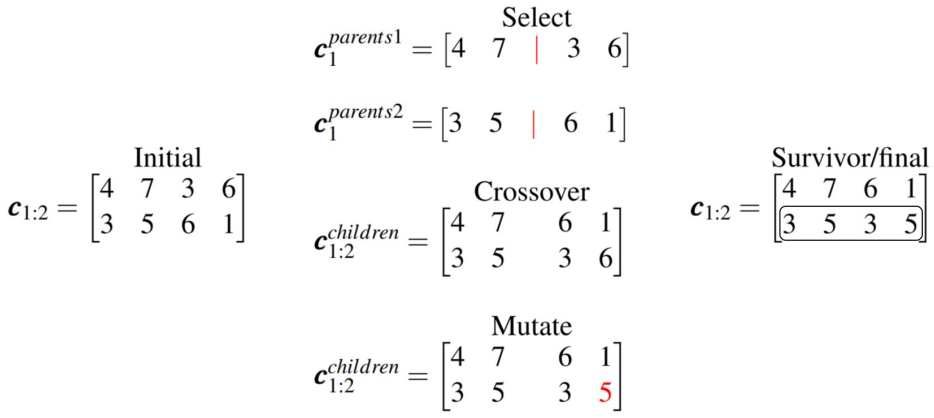


Figure 4.16 Example of evolution of the population $\boldsymbol{c}_{1:M}$ with only two individuals ($M = 2$) in a MTS with the 3x3 belief map of Figure 4.13, a single UAV and a decision horizon of $N = 4$ values. The survivor individual with the optimal solution $c_1^{1:4} = \{3,5,3,5\}$ is highlighted.

As an illustrative example of how GA obtains its values, we will reuse the example of Figure 4.13. Figure 4.16 shows the evolution of a population with only two individuals in one iteration of GA. The displayed example supposes that in the selection step, both solutions are selected as parents, crossed by the randomly selected middle point to obtain the two children and then the second child has its last gene mutated to one of its surrounding cardinal directions. Next, the survivors are selected among the best individuals within the original $\boldsymbol{c}_{1:2}$ and new population of solutions $\boldsymbol{c}_{1:2}^{children}$, creating the final set of survivor individuals, which already contains the optimal sequence of actions $c_1^{1:4} = \{3,5,3,5\}$. It is worth noting that although for this example we have selected the crossover point and mutation genes that lets us find the optimal solution in only one iteration, any other crossover point or mutation could be performed, and thus the algorithm would require several iterations to find the optimal solution.

4.3.4.2.2. Summary of the Properties of the MTS Algorithms. In this section we analyze the main conceptual similarities and differences between the analyzed algorithms attending to the criteria described below and summarized in Table 4.3.

Algorithms	Direct solution manipulation	Probability learning	Dependency Exploitation	Ad-hoc MTS Heuristic
CEO		✓		
BOA		✓	✓	
GA	✓			
MMAS-NODE+H		✓		✓
MMAS-TIME+H		✓		✓

Table 4.3 Summary of the most relevant properties of each algorithm.

The subsequent analysis of the algorithms performance will allow to deduce which of these properties are more advantageous for MTS.

Manipulating directly the solutions of the algorithm to construct new ones (second column of Table 4.3). This characteristic is exploited only by GA, which treats directly with populations of solutions whose fitness gets better iteration by iteration through the direct manipulation of individuals and discarding solutions with worse fitness values.

Learning/storing probability distribution/information to sample new solutions according to the best solutions identified in previous iterations of the algorithms (third column of Table 4.3). On the one hand, CEO and BOA sample their solutions from the estimated probability distribution learned using the best solutions found by the algorithms. On the other hand, MMAS samples new solutions from a probability distribution constructed using the pheromone table, which contains information about the best actions taken at each node or time step by the best solutions of previous algorithm iterations. The only algorithm that does not fit this criterion is the GA-based MTS algorithm, because although some GA implementations include probabilistic learning mechanisms (e.g. in Thierens and Bosman (2011)), in general this approach (and in particular the implemented one) does not learn a probability distribution.

Exploiting the dependencies among the values of the decision variables while learning the probabilities (fourth column of Table 4.3). This property is only ex-

ploited by BOA and our analysis will show if this computer costly operation is beneficial to determine the optimal solution.

Employing a constructive heuristic especially designed for the MTS problem in the generation of the solutions of the algorithm to help it to identify sooner promising solutions (fifth column of of Table 4.3). This property is only used in the MMAS-based algorithms (MMAS-NODE+H and MMAS-TIME+H) proposed in this thesis and the analysis should corroborate if this property speeds up the location of good solutions or makes the algorithm converge too early to local solutions. It is worth clarifying that MTS specific information could be also considered by the other optimization techniques by including a percentage of heuristic solutions (constructed considering uniquely a MTS heuristic) in some populations of solutions. In this way, an initial population with heuristic solutions would usually permit the approach to achieve initially better fitness values than from a completely randomly generated population. And the inclusion of heuristic solutions (also known as immigrants) in some generations would incorporate problem specific knowledge at different iterations of the algorithm. However, it is worth noting the difference between this approach (commonly used in GA based algorithms) and the one followed by MMAS, where the heuristic information is combined with the information learned from previous iterations (saved in the pheromone table) and used during all the algorithm iterations.

4.3.4.2.3. Comparative Results with other Algorithms. In this section we compare the performance of one of the proposed MTS algorithm (MMAS-NODE+H) with other state of the art MTS algorithms (based on CEO, BOA and GA) over the search scenarios described in Section 4.3.1.

All the considered algorithms have a population size proportional to the scenario complexity ($R = 8 \cdot N \cdot U$) and consider a maximum predefined computational time as stop condition. It is worth mentioning that we have also performed a statistical analysis over different parameterizations of CEO, BOA and GA based algorithms in order to make a fair comparison against good versions of all of them. The specific

parameters of each optimization technique are set (after the statistical analysis) as follows. The CEO smoothing parameter is set to the value proposed in (Lanillos et al., 2012) for the MTS problem ($\gamma = 0.6$). In the case of BOA, after performing a statistical comparison of different BN learning strategies for the MTS problem, we have substituted the one used in (Lanillos et al., 2013) by the K2 algorithm proposed in (Cooper and Herskovits, 1992) that assumes an ancestral ordering among the variables[2] as this strategy showed faster convergence to similar ET values. Besides, for CEO and BOA we make the percentage of solutions used to learn the probability distribution $\varepsilon = 0.1$. For GA we have selected a crossover probability of $p_{xover} = 0.8$ and a probability of $p_{mut} = 1/(8 \cdot N)$ that a gene (solution component) of the offspring population is modified to one of its surrounding cardinal actions.

Figure 4.17 shows the ET and dominance evolution graphs of the comparative analysis over the six search scenarios. On the one hand, the dominance graphs, located in the top graphics of each scenario, show the dominance relationships of the different MTS algorithms against MMAS-NODE+H, selected as the base algorithm. These graphs show that the proposed ant colony based algorithm outperforms the other algorithms in all scenarios. On the other hand, the ET evolution graphs show that MMAS-NODE+H finds much better solutions at the first iterations and reaches faster higher quality solutions than the other algorithms thanks to the use of the MTS heuristic. Additionally, the ET evolution graphs show that GA presents better results than CEO and BOA. And although BOA outperforms CEO in Scenario C, it has the slowest convergence and requires a high computational cost due to the time required for learning the BN structure. More concretely, the computational time required for ten iterations of BOA (observed by the time difference between two consecutive dots) ranges from 3 to 20 seconds depending on the scenario. Finally, despite that due to the computation of MTS heuristics MMAS-NODE+H requires a higher computational time for each iteration than CEO and GA, the use of the MTS heuristics allows the proposed algorithm to find higher quality solutions faster than all the other algorithms.

[2]This implies that the variables (cardinal control actions taken by UAVs) can only depend on the variables that appear earlier in the proposed ordering (cardinal actions taken in previous steps).

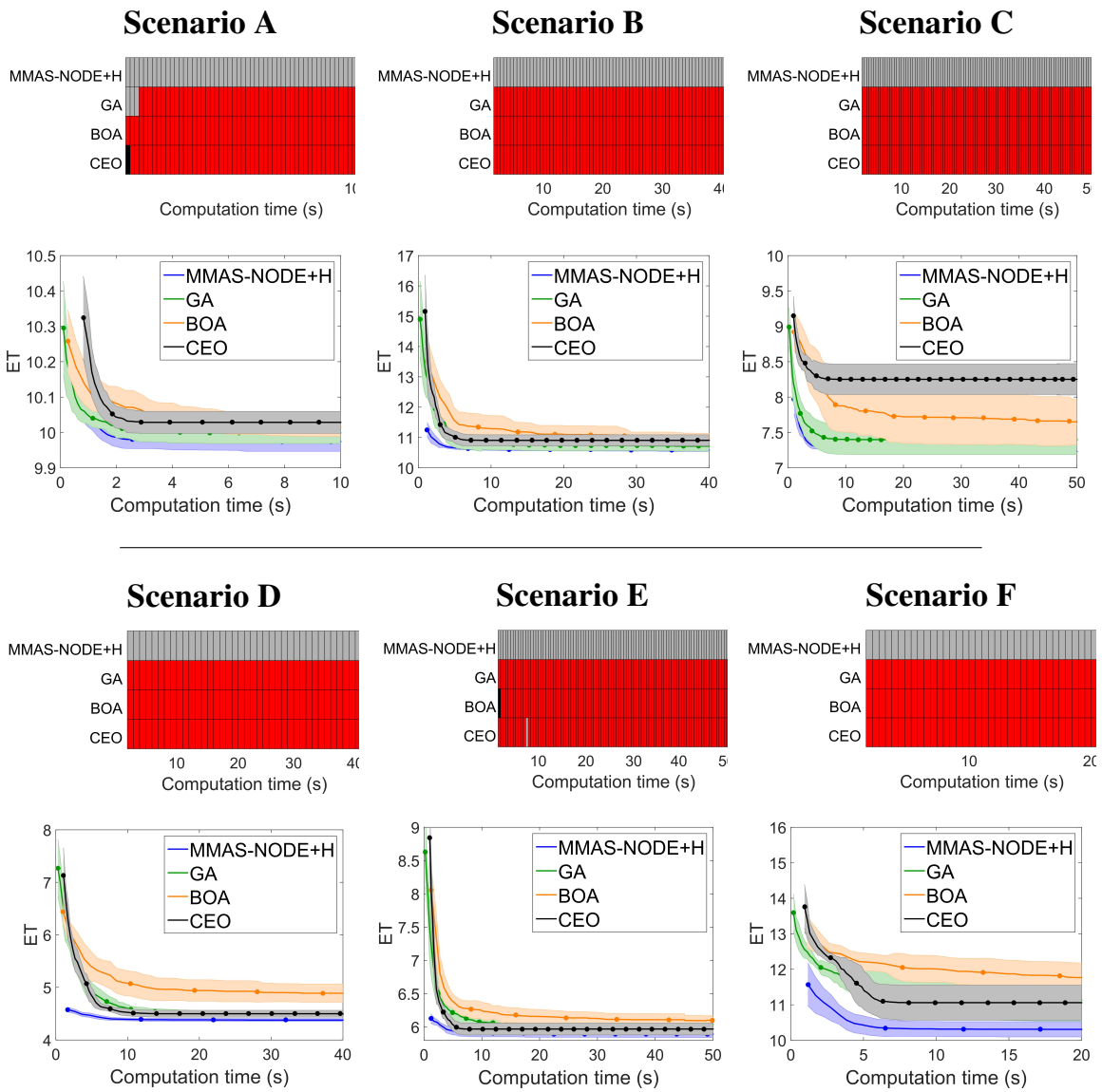


Figure 4.17 Comparison of different MTS algorithms. The base variant in the dominance evolution graphs is MMAS-NODE+H.

4.3.5. Summary

This chapter proposes two MTS algorithms based on ant colony optimization techniques, which optimize the search trajectories of a group of UAVs looking for a lost target in an uncertain environment. The chapter starts by describing the approach followed to model the problem that considers an ideal sensor model, a cardinal UAV dynamic model and the expected target detection time as fitness criterion. Neverthe-

less, if required, the proposed algorithms can be used straightforward with different sensor models (e.g. radars), extended to a multi-objective optimization or used with other discrete motion models (e.g. cardinal motion model with a maximum turning angle restriction). The limitation of the algorithms to discrete UAV motion models is imposed by the use of a discrete optimization technique (MMAS) that requires discrete decision variables.

The chapter also details how the MTS problem is formulated in order to optimize it using MMAS. To this end, the algorithm solutions (UAV search trajectories) are codified as sequences of cardinal actions that MMAS sequentially constructs combining the information of a MTS heuristic and the information learned from previous iterations (saved in a pheromone table). The proposed MTS heuristic guides the UAVs toward the closer and higher probability areas of the belief, favoring in this way solutions with low expected time. Besides, we propose two different pheromone encodings: learning the best actions to take at each cell (in algorithm MMAS-NODE+H) or learning the best actions to take at each time step (in algorithm MMAS-TIME+H).

The performances of the proposed algorithms are analyzed and compared with previous state of the art algorithms over six search scenarios, which differ on the target initial belief and dynamics, number of UAVs and planning horizon. The first part of the analysis focuses on the proposed approach: analysing the performance of different parameter settings, studying the power of the MTS heuristic and of the pheromone learning mechanism, and comparing the performance of the two encodings. From this analysis we can conclude that the two proposed algorithms obtain high quality solutions. Besides, on the one hand, the MTS heuristic allows the algorithms to obtain higher quality solutions from the first iteration and converge faster to higher quality results. On the other hand, the pheromone mechanism also contributes in the performance of the two proposed algorithms, allowing its use to reach lower expected times in all the scenarios. The second part of the analysis compares our ant colony based approach against different MTS heuristic strategies proposed in (Meghjani et al., 2016) and several optimization methods (based on CEO, BOA and GA) previously used for MTS. From this analysis we can conclude that the ant colony ap-

proach presented in this thesis outperforms the other MTS algorithms, mainly thanks to the use of the proposed MTS heuristic that allows the algorithm to converge faster to higher quality solutions.

To sum up, this chapter presents two MTS algorithms based on ant colony optimization techniques that obtain high quality search routes (as sequences of adjacent cells that the UAVs have to overfly) combining the information learned from promising solutions identified in the previous algorithm iterations and information from a heuristic specifically designed for MTS. The use of this MTS heuristic allows the algorithms to obtain higher quality solutions from the first iterations and to converge faster to solutions with lower expected times of finding the target. This is a great advantage in a high-complex problem like MTS, where it is necessary to reach a good balance between the quality of solutions and the computational time.

Chapter 5

Multi-criteria MTS Algorithms for Continuous UAV Motion Models

"Divide et vinces"

Julius Caesar

This chapter presents the proposed multi-criteria MTS algorithm with a continuous UAV dynamical model and a realistic sensor model. Contrary to the MTS algorithms for discrete UAV dynamic models presented in Chapter 4, the algorithms presented in this chapter consider continuous UAV dynamical models that fulfill fixed-wing dynamic restrictions. Besides, the algorithm considers a realistic sensor model and optimizes multiple criteria and constraints. Due to the successful performance of ACO techniques in the discrete approach, in this chapter we select a continuous ant colony based algorithm and propose a continuous MTS heuristic to test if it allows to reduce the computational time (now higher due to the complexity added by the UAV models and by the evaluation criteria).

The chapter is organized as follows. First, we describe the selected UAV models, the codification of the solutions and the multi-criteria evaluation methodology. And next, we introduce the proposed MTS algorithm based on the continuous ant colony based technique ACOR. Finally, the performance of the proposed MTS algorithm is

analyzed over several search scenarios and compared with a MTS algorithm based on Genetic Algorithms (GA).

5.1. MTS Continuous Approach

In MTS algorithms with continuous models the UAV trajectories are codified as a sequence of continuous variables (commanded control variables) and are no longer restricted to the sequence of cells of the search region to overfly by the UAVs. However, it is important to clarify that the discretization of the belief and target dynamics is maintained. Besides, in this chapter we also include a new sensor model, whose behavior changes continuously with the distance between the UAV and the target. However, the approaches in this chapter could be used with any sensor model (including the one used in the previous chapter). This section presents the followed approach: the selected UAV models (continuous dynamic model and sensor model) and the problem formulation (codification of solution and evaluation criteria).

5.1.1. UAV Models

This section describes the selected realistic UAV models for the continuous approach: a non-linear dynamic model implemented in Simulink and a downward primary looking radar model for the UAV sensors.

5.1.1.1. UAV Dynamic Model

The selected UAV motion function $\dot{s}_u = f(s_u^t, c_u^t)$ is defined in the non-linear Simulink model represented in Figure 5.1. The variables within the UAV state s_u^t, highlighted in light green at the right of the figure, are the UAV 3D location (x_u^t, y_u^t, h_u^t), speeds $(\dot{x}_u^t, \dot{y}_u^t, \dot{h}_u^t)$, heading (θ_u^t), course angle $(\theta_{u,course}^t)$, air velocity (v_u^t), ground velocity (v_u^t), and fuel consumption $(fuel_u^t)$. The variables within the control signal c_u^t, highlighted in cyan on the left of the figure, are the commanded airspeed $(v_u^{c,t})$, commanded heading $(\theta_u^{c,t})$ and commanded height $(h_u^{c,t})$. The environment in-

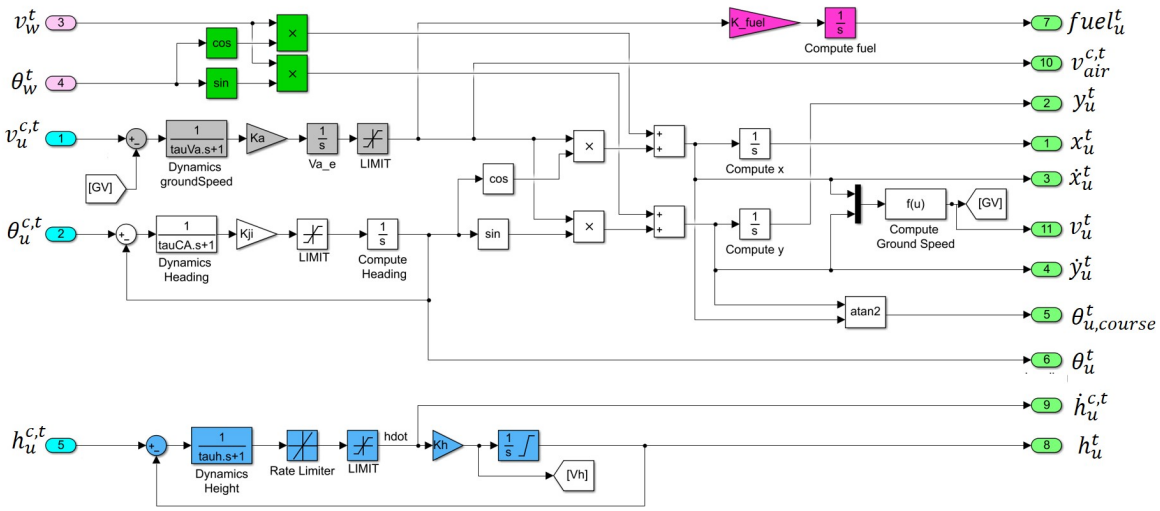


Figure 5.1 UAV dynamic model implemented in Simulink.

fluence is defined by the wind speed (v_w^t) and direction (θ_w^t), highlighted in pink. In case that the wind speed is zero, the heading coincides with the course angle and the air velocity with the ground velocity. Besides, blue colored blocks in Figure 5.1 are used for modeling the height dynamics, green for the wind, magenta for the fuel, gray for the air velocity and white for the lateral dynamics. The model also includes the usual limitations related with the air velocity, height, and heading. The use of this UAV dynamic model and its integration with Simulink allows our MTS algorithms to produce feasible search trajectories that meet the dynamic restrictions of fixed-wing UAVs.

Moreover, the behavior of the Simulink model can be tuned by means of a parameterization file, modifiable and selectable by the user, that is automatically loaded before each simulation. This allows to quickly adapt the model to the flying constraints of different types of fixed-wings UAVs (and if required, to include different types of UAVs in the same MTS scenario).

5.1.1.2. Sensor Model

The considered sensor is a downward primary looking radar that detects the signal returned by the searched target and considers a detection measurement when the

power of the returned signal is over a threshold. Its likelihood $P(D_u^t|v^t,s_u^t)$ determined by Equation 5.1 and derived from (Budge, 2011) has previously been used for MTS in (Lanillos et al., 2014a; Pérez-Carabaza et al., 2016b) and (Pérez-Carabaza et al., 2017). The probability of target detection has a non-linear dependency with the Signal to Noise Ratio (S_{NR}) and Threshold to Noise Ratio (T_{NR}), defined respectively as the ratio of the power of the signal/threshold by the power of the noise.

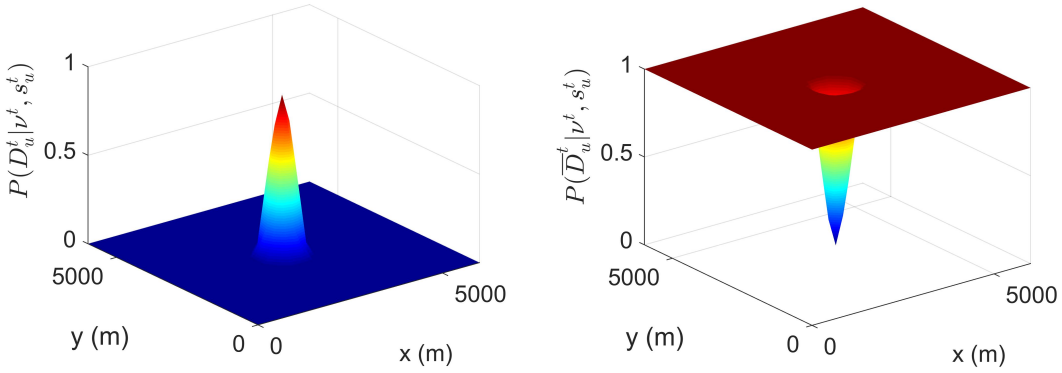
$$P(D_u^t|v,s_u^t) = \left(1+\frac{2\cdot S_{NR}\cdot T_{NR}}{(2+S_{NR})^2}\right)e^{-2\cdot T_{NR}/(2+S_{NR})} \tag{5.1}$$

$$T_{NR} = -\log(P_{fa}) \tag{5.2}$$

$$S_{NR} = C_\varepsilon/(d_{v,u}^t)^4 \tag{5.3}$$

The threshold to noise ratio T_{NR} is determined by the probability of false alarm P_{fa} with Equation 5.2, where P_{fa} is the probability of having a false target detection. Each time a detection measurement is obtained, the radar data processor has to verify the detection, consuming time and energy, so low false alarm probability values are desired. However, low P_{fa} values increase the values of T_{NR}, what has the undesired effect of decreasing the probability of detection. Therefore, generally an acceptable P_{fa} value is considered in order to define the threshold. Equation 5.3 states that the signal to noise ratio S_{NR} decreases exponentially with the euclidean distance $d_{v,u}^t$ between the sensor/UAV 3D location (x_u^t, y_u^t, h_u^t) and the target location (x_v^t, y_v^t, h_v^t). The constant C_ε includes multiple characteristics of the radar (such as the radar wavelength and cross-section) and can be indirectly determined from Equations 5.1-5.3 by fixing the detection likelihood $P(D_u^t|v^t,s_u^t)$ for a given distance $d_{v,u}^t$ and probability of false alarm P_{fa}.

Figure 5.2 displays the radar likelihood parametrized considering $P_{fa} = 10^{-6}$ and $P(D_u^t|v^t,s_u^t) = 0.9$ at $d_{v,u}^t = 250$ m for a UAV located at the center of the search area at an altitude of 300 m. The probability of detecting the target $P(D_u^t|v^t,s_u^t)$, displayed on Figure 5.2 (a), has is maximum under the UAV location and decreases as the distance to the target increases. On the contrary, its complementary probability of



(a) $P(D_u^t|v^t,s_u^t)$ (b) $P(\overline{D}_u^t|v^t,s_u^t)$

Figure 5.2 Radar likelihood for $P(D_u^t|v^t,s_u^t)=0.9$ at $d_{v,u}^t=250$ and a probability of false alarm $P_{fa}=10^{-6}$ for a UAV located at the center of the search area $(3000,3000,300)$.

non target detection $P(\overline{D}_u^t|v^t,s_u^t)$, displayed on Figure 5.2 (b), has its minimum value under the UAV location and increases with the distance.

We have selected this sensor model to test the continuous MTS algorithms due to its realistic smooth distance-decaying shape, common to other UAV sensors.

5.1.2. Continuous MTS Formulation

In order to formulate the MTS as a multi-objective optimization problem we have to define a way of codifying the decision variables within the optimization algorithm and the fitness functions that have to be optimized. Moreover, in order to tackle the high-complexity of the MTS problem, we solve several smaller problems (each associated to a different sequence of the trajectory) following a multi-stepped (receding horizon control) approach.

5.1.2.1. Codification of the Decision Variables

Analogously to the discrete MTS algorithms the solutions are codified in the action space $c_{1:U}^{1:N}$ and then pass through the UAV dynamic model in order to obtain the UAVs search routes $s_{1:U}^{1:N}$. However, in the continuous approach the solutions

are codified as sequences of continuous decision variables; the commanded decision variables of the UAV dynamical model that are applied during a prefixed time interval. Currently our algorithm only determines the best commanded heading sequence $\theta_{1:U}^{c,1:N}$, by pre-fixing the commanded values of the UAVs velocity $v_{1:U}^{c,1:N}$ and height $h_{1:U}^{c,1:N}$ during the whole search mission. In other words, the solutions that the proposed algorithm optimize are sequences of UAV commanded headings:

$$\overbrace{\theta_1^{c,1}, \theta_1^{c,2}, ..., \theta_1^{c,N}}^{\text{UAV 1}}, \overbrace{\theta_2^{c,1}, \theta_2^{c,2}, ..., \theta_2^{c,N}}^{\text{UAV 2}}, ..., \overbrace{\theta_U^{c,1}, \theta_U^{c,2}, ..., \theta_U^{c,N}}^{\text{UAV U}} \tag{5.4}$$

5.1.2.2. Evaluation Criteria

In order to evaluate a sequence of command control inputs of the U UAVs (optimized headings $\theta_u^{c,l:N}$ at constant speed v_u^c and altitude h_u^c) applied during fixed intervals of time ΔT, we use the UAV motion model implemented in Simulink to calculate the state s_u^j of each UAV at J evenly spaced time steps j. Then, the search trajectories of the UAVs $s_{1:U}^{0:N}$ are evaluated according to two feasibility criteria (collision and NFZ avoidance) and two performance criteria (expected time and a myopia heuristic reduction criterion) described below. Before, note that the state variable superindex is different in the constraint expressions (s_u^j) and in the objective functions (s_u^t) in order to let the reader distinguish between the time step discretization used in each case (since for evaluating the constraints we often need a smaller time step than for evaluating the objective functions).

Feasibility Objectives. Although the UAV trajectories obtained from the sequence of control signals are already feasible from the UAV maneuverability point of view, the algorithm has to check if the UAVs may collide between them or fly over forbidden regions (NFZ). To do it, it minimizes the number of basic time steps j that the UAVs are within any NFZ (Equation 5.5) or that they do not maintain a security distance (Equation 5.6).

$$\#_{\text{NFZ}} = \sum_{j=1}^{J}\sum_{u=1}^{U} WithinNFZ(x_u^j, y_u^j) + CloseNFZ(x_{1:U}^J, y_{1:U}^J, \theta_{1:U}^J) \tag{5.5}$$

$$\#_{\text{COL}} = \sum_{j=1}^{J}\sum_{u=1}^{U}\sum_{k=1+u}^{U} Collision(x_u^j, y_u^j, h_u^j, x_k^j, y_k^j, h_k^j) \tag{5.6}$$

where *WithinNFZ* is a function that checks if a UAV located at (x_u^j, y_u^j) is inside any of the NFZs and *Collision* is a function that checks if the position of a UAV (x_u^j, y_u^j, h_u^j) is closer than a security distance from the location of other UAV located at (x_l^j, y_l^j, h_l^j). In addition, the function *CloseNFZ* of Equation 5.5 adds a penalizing term if any of the UAVs, starting from their final positions given by $(x_{1:U}^J, y_{1:U}^J, \theta_{1:U}^J)$, will not be able to avoid the NFZ during its future movements.

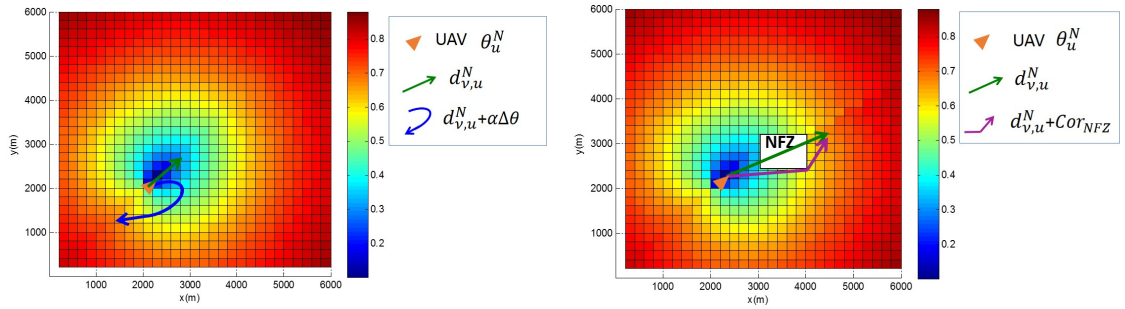
Expected Time. We optimize the expected value of the target detection time (ET) of the search trajectories, which is obtained with Equation 5.7, adding up at each time step the "unnormalized belief", which is recursively obtained with Equation 5.8 updating $\tilde{b}(\nu^0)$ (given by Equation 5.9) with the target dynamic information $P(\nu^t|\nu^{t-1})$ and sensor measurements $P(D_u^t|\nu^t, s_u^t)$ from time step 0 to time step t.

$$ET(s_{1:U}^{1:N}) = \sum_{t=0}^{N}\sum_{\nu^t \in G_\Omega} \tilde{b}(\nu^t) \tag{5.7}$$

$$\tilde{b}(\nu^t) = \sum_{\nu^t \in G_\Omega}\prod_{u=1:U}(1 - P(D_u^t|\nu^t, s_u^t))\sum_{\nu^{t-1}\in G_\Omega} P(\nu^t|\nu^{t-1})\tilde{b}(\nu^{t-1}) \tag{5.8}$$

$$\tilde{b}(\nu^0) = \prod_{u=1:U} P(\overline{D}_u^0|\nu^0, s_u^0) b(\nu^0) \tag{5.9}$$

Myopia Heuristic Reduction Criterion. We consider an additional performance criterion to tackle the algorithm myopia caused when the optimization of the full trajectory is divided in Q optimization steps of smaller planning horizon ($L = N/Q$) following a receding horizon control approach. The proposed myopia heuristic reduction criterion measures the incapability of collecting the remaining belief from the ending observation points at the optimization step q of the UAVs trajectories $s_{1:U}^{qL}$.



(a) $H(v^{qL}, s_u^{qL})$ corrected by the UAV disorientation.

(b) $H(v^{qL}, s_u^{qL})$ corrected by a NFZ presence $P(\overline{D}_u^t | v^t, s_u^t)$.

Figure 5.3 Myopia heuristic reduction criterion sketch.

It is calculated with Equation 5.10, weighting the remaining $\tilde{b}(v^{qL})$ by a monotonically increasing function $H(v^{qL}, s_u^{qL})$ with the growth of $F(v^{qL}, s_u^{qL})$, which reaches a minimum value of $H(v^{qL}, s_u^{qL}) = 0$ when $F(v^{qL}, s_u^{qL}) = 0$ and a maximum value of $H(v^{qL}, s_u^{qL}) = 1$ when $F(v^{qL}, s_u^{qL}) = \infty$.

$$\text{MYOP} = \sum_{v^{qL} \in G_\Omega} \prod_{u=1}^{U} H(v^{qL}, s_u^{qL}) \tilde{b}(v^{qL}) \tag{5.10}$$

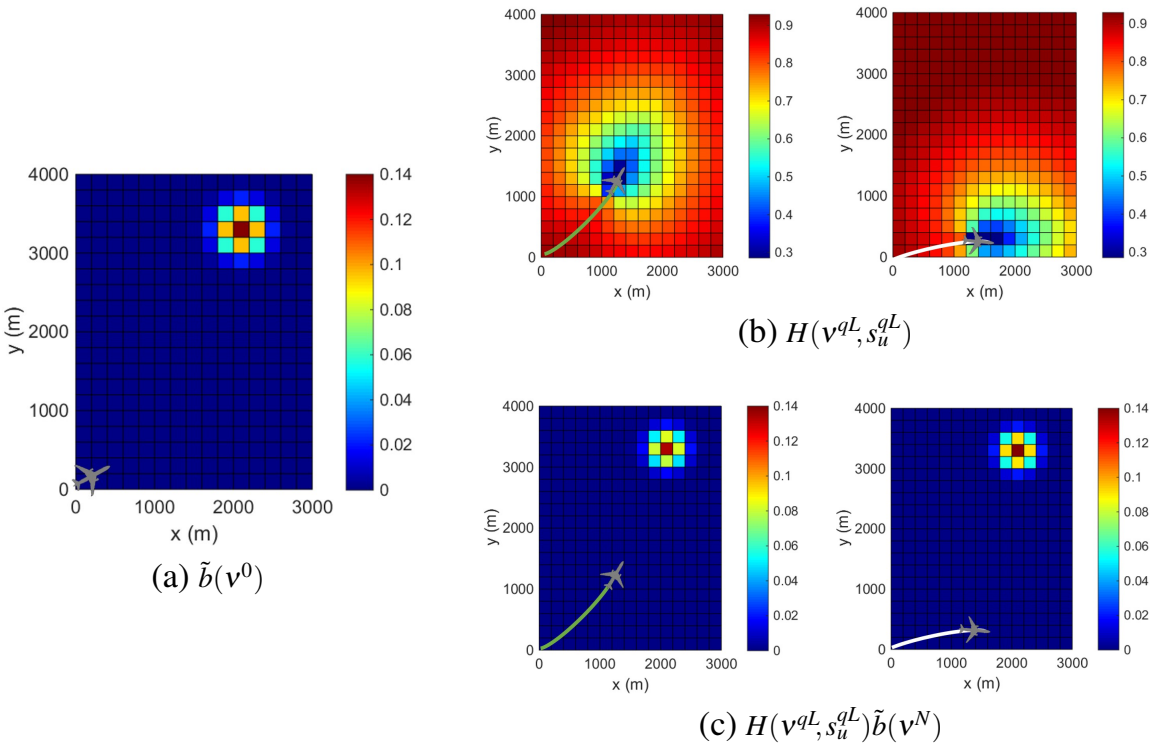
$$H(v^{qL}, s_u^{qL}) = 1 - \lambda^{F(v^{qL}, s_u^{qL})} \text{ with } 0 < \lambda < 1 \tag{5.11}$$

$$F(v^{qL}, s_u^{qL}) = d_{v,u}^{qL} + \Upsilon \left| \theta_u^{qL} - \arctan \frac{v_y - y_u^{qL}}{v_x - x_u^{qL}} \right| + Cor_{NFZ} \tag{5.12}$$

Moreover, $F(v, s_u^{qL})$ is an extended distance function that gives higher values to the locations of the belief v^{qL} that are further from the UAV reach. Its value is obtained with Equation 5.12 where the first summation term accounts for the length $d_{v,u}^{qL}$ of the straight line that joins the u-th UAV N-th location with possible target location v^{qL}, the second term $\Upsilon \left| \theta_i^{qL} - \arctan \frac{\tau_y - y_i^{qL}}{\tau_x - x_i^{qL}} \right|$ accounts for the extra-distance needed (due to the UAV dynamics) to reorient the UAV heading θ_i^{qL} towards the straight line that joins the UAV and target location, and the last correction term Cor_{NFZ} is related with the NFZ locations with respect to s_u^{qL}.

The effects of the terms in $F(v^{qL}, s_u^{qL})$ over $H(v^{qL}, s_i^{qL})$ are sketched in Figure 5.3, where the UAV location and orientation are indicated by the orange triangle. The absence of NFZ in Figure 5.3 (a) makes $H(v^{qL}, s_u^{qL})$ equal to 0 in the cell under the final location s_u^N and grow asymptotically towards 1 with the increment of the distance between the UAV and each cell and of the discrepancy between the orientations of the UAV and the straight line that joins the UAV location with each cell. The NFZ presence in Figure 5.3 (b) is accounted by Cor_{NFZ} that considers the extra distance/reorientation needed to make the UAV avoid the NFZ. Besides, the parameter λ controls the importance of further probability areas: higher λ values make $H(v^{qL}, s_u^{qL})$ increase slower with $F(v^{qL}, s_u^{qL})$, while lower λ values make $H(v^{qL}, s_u^{qL})$ have a sharper increase giving similar weights (slightly smaller than 1) to further cells. We adjust the value of λ considering the possible future *range* of the UAV (cells that can be reach in future optimization steps) taking into account the number of optimization steps left in order to give similar weights (close to 1 and thus without influence) to the cells that will be further from the UAV reach in the remaining optimization steps (i.e. $\lambda = (1-0.95)^{(1/range)}$, Equation 5.11). In the case that there are no further sequences to optimize, the possible range at the UAV final state s_u^N is zero, so $\lambda = 0$ and $H(v^N, s_u^N)$ has a constant value of 1, as the adequacy of s_u^{qL} for future optimization steps is irrelevant at the last optimization step of the planner.

We use the example of Figure 5.4 to illustrate how the myopia heuristic reduction criterion helps to avoid myopic situations. Figure 5.4 (a) shows the initial belief $b(v^0)$ and UAV location, and Figures 5.4 (b) and (c) display the functions ($H(v^{qL}, s_u^{qL})$ and $H(v^{qL}, s_u^{qL})\tilde{b}(v^{qL})$) used to calculate the myopia heuristic reduction criterion corresponding to two different trajectories obtained in the first optimization step of a total of 3 optimization steps. Besides, although both trajectories gathered null probability (and therefore have the same ET), the final location of the green one (on the left side) makes $H(v^{qL}, s_u^{qL})$ weight the values of $\tilde{b}(v^{qL}) \neq 0$ with lower values than the final location of the white one (on the right side). Hence, the value of the myopia heuristic reduction criterion (computed with Equation 5.10 adding up the values of all the cells of the product of $H(v^{qL}, s_1^{qL})$ and $\tilde{b}(v^{qL})$) of the green trajectory ($MYOP = 0.84$) is



(a) $\tilde{b}(v^0)$

(b) $H(v^{qL}, s_u^{qL})$

(c) $H(v^{qL}, s_u^{qL})\tilde{b}(v^N)$

Figure 5.4 Heuristic example sketch. The myopia heuristic reduction criterion of the non myopic trajectory displayed in green $\text{MYOP} = \sum_{v^{qL} \in G_\Omega} H(v^{qL}, s_1^{qL})\ \tilde{b}(v^{qL}) = 0.84$ is smaller than the one of the myopic trajectory displayed in white $\text{MYOP} = \sum_{v^{qL} \in G_\Omega} H(v^{qL}, s_1^{qL}) \tilde{b}(v^{qL}) = 0.95$.

smaller than the one of the trajectory in white ($MYOP = 0.95$). Therefore, the minimization of $MYOP$ criterion helps to avoid myopic solutions by assigning smaller values to UAVs trajectories whose final positions are closer and better oriented to the remaining high probability areas.

Furthermore, we want to remark the differences between our myopia strategy and the heuristic sensorial function introduced in (Lanillos et al., 2014b), which served as inspiration to ours. On the one hand, the function $H(v, s_u^{qL})$ in (Lanillos et al., 2014b) only includes the first term in Equation 5.12 and therefore, it does not account as our $H(v^{qL}, s_u^{qL})$ for the distance corrections needed to reorientate the UAV or avoid NFZ. On other hand, the parameter λ is fixed and not adjusted in order to give influence only to the areas that are still reachable. Furthermore, the function in (Lanillos et al., 2014b) is used as an additional terminal infinitive range sensor within the probability

of no detection criterion, instead of as a new optimization criterion correlated and complementary to ET.

It is also worth noting the relationship between the myopia optimization criterion MYOP and the expected detection time ET. As Equation 5.10 states, MYOP weights the unnormalized remaining target belief $\tilde{b}(v^{qL})$ obtained when the target is not detected at the observation steps of the UAVs trajectories, with the product of the heuristic $H(v^{qL}, s_u^{qL})$ evaluated at the final positions of the trajectory of each UAV. Therefore, the value of MYOP is reduced 1) when the UAVs gather target probability overflying regions where there are probability of target presence and 2) when the final locations of the UAVs are near and well oriented towards regions where the belief is concentrated. Although the first type of reduction also modifies the value of ET, the ET is also influenced by the order in which the cells of the search regions were observed (i.e. the sooner the probability is collected the better), while MYOP is not. For that reason, ET and MYOP are simultaneously correlated and complementary: both benefit from visiting cells of the grid with high probability of target presence, ET from a visiting order that collects the belief as quicker and possible, and MYOP from distributing the final locations of the UAVs to let them arrive, in the future, as soon as possible to cells with high probability values.

5.1.2.3. Multi-stepped Approach

Our MTS algorithm, which determines the sequence of N commanded headings of all the UAVs ($\theta_{1:U}^{c,1:N}$) that minimizes the Feasibility and Optimization Criteria array FOC $= [\#_{\text{NFZ}}, \#_{\text{COL}}, \text{ET}, \text{MYOP}]$, is implemented with the multi-step receding horizon control approach presented in Algorithm 8. This approach divides the whole sequence of actions N in Q subsequences of length $L = N/Q$ and uses the ant colony based algorithm ACOR (which will be later presented in Algorithm 10) to optimize each action subsequence from the ending state ($s_{1:U}^{qL}$ and $\tilde{b}(v^{qL})$) of the previously obtained action subsequence. This way of proceeding, supported by the time-additive nature of the optimization objectives (Equations 5.5, 5.6 and 5.7), lets the algorithm handle the complexity of the problem by iteratively finding the solutions of smaller

Algorithm 8 Multi-stepped MTS Algorithm

Require: $b(\nu^0), P(\nu^t|\nu^{t-1}), P(D_u^t|\nu^t, s_u^t)$ ▷ Initial target belief and target motion model
Require: $s_{1:U}^0$ ▷ Initial UAVs location
Require: N, L ▷ Whole and subsequence planning horizons
1: $t \leftarrow 0$ ▷ Time starting index
2: $\tilde{b}(\nu^t) \leftarrow \prod_{u=1:U} P(\overline{D}_u^0|\nu^0, s_u^0)P(\nu^0)$ ▷ Initialize "unnormalized belief"
3: FOC^t=[0,0,0,0] ▷ Initialize FOC array
4: $\mathbf{s}^{gb} \leftarrow s_{1:U}^0$ ▷ Initialize the global best solution with initial UAVs states
5: $Q = N/L$ ▷ Total number of subsequences
6: **for** q=1:Q **do** ▷ For each subsequence
7: $[s_{1:U}^{t:t+L}, \text{FOC}^{t+L}] \leftarrow ACOR(\tilde{b}(\nu^t), s_{1:U}^t, t, L, \text{FOC}^t)$
8: $\mathbf{s}^{gb} \leftarrow \{\mathbf{s}^{gb}, s_{1:U}^{t+1:t+L}\}$ ▷ Add the new step of the search trajectory
9: $\tilde{b}(\nu^{t+L}) \leftarrow UpdateBelief(\tilde{b}(\nu^t), P(\nu^t|\nu^{t-1}), P(D_u^t|\nu^t, s_u^t), s_{1:U}^{t:t+L})$
10: $t \leftarrow t + L$
11: **end for**
12: **return** $\mathbf{s}^{gb}, FOC^N$ (Complete sequence of actions and its evaluation criteria)

problems. Besides, the myopia heuristic reduction criterion (MYOP given by Equation 5.10) helps to mitigate the myopic effects derived from the multi-stepped approach.

The procedure requires the probability models (target initial belief $b(\nu^0)$, target dynamic model $P(\nu^t|\nu^{t-1})$ and sensor model $P(D_u^t|\nu^t, s_u^t)$), the initial UAV locations ($s_{1:U}^0$), the planning horizon of the whole trajectory (N) and the planning horizon of each subsequence (L). At the beginning, the procedure initializes the time index (in line 1), the "unnormalized belief" (in line 2), the FOC array (in line 3), and the global best trajectory solution $\mathbf{s}^{gb}$ (in line 4). Besides, in line 5 the total number of steps of the receding horizon controller approach (Q) are computed dividing the planning horizon N by the the planning horizon of each subsequence L. Within the receding horizon controller loop (from lines 6 to 11), each subsequence its optimized and the returned trajectory subsequence is concatenated with the previous ones and saved in $\mathbf{s}^{gb}$. Besides, the "unnormalized belief" is recursively updated by the *UpdateBelief* function (in line 9) according to the target movements ($P(\nu^t|\nu^{t-1})$) and

sensor measurements ($P(D_u^t|v^t, s_u^t)$ and $s_{1:U}^{t:t+L}$) with Equation 5.8. In this way, the next optimization step considers the resulting "unnormalized belief" from the previous optimization steps. Finally, once the Q optimization steps have finished, the procedure returns the complete search trajectory and its corresponding fitness array.

5.2. MTS-ACO Continuous Approach

In this section we propose a MTS algorithm supported by the continuous ant colony algorithm ACOR (Socha and Dorigo, 2006). First, we present a background of ACO techniques for continuous optimization problems. Next, we describe the mechanisms that ACOR use to save the information learned from previous iterations and how new solutions are sampled from it. Besides, we propose a continuous MTS heuristic and a mechanism to include heuristic information in ACOR through the use of heuristic ants. Finally, the general pseudocode of the proposed algorithm is presented.

5.2.1. Introduction to ACO in Continuous Domains

Ant colony based algorithms like AS (Dorigo et al., 1996) or MMAS (Stützle and H. Hoos, 2000) have been successfully applied to a variety of discrete optimization problems like TSP, where each solution component has associated a limited set of possible values. However, these techniques are not appropriate to optimize problems that involve continuous variables. In contrast to previous discrete ACO algorithms, ACOR (Socha and Dorigo, 2006) is an ant colony based algorithm designed to solve problems with continuous decision variables that have associated a continuous range of possible values.

Although many real world optimization problems may be represented as discrete optimization problems in a straightforward way, there exists a large number of problems whose decision variables have to be chosen from a continuous range. Continuous optimization problems can be solved with discrete optimization techniques

by discretizing the range of allowed values of the continuous variables to a finite set. However, if the resolution required or the variables range is high, this implies a large set of values and this approach may not be convenient. For this reason, some optimization techniques like AS, MMAS or Greedy Randomized Adaptive Search Procedure (GRASP, (Feo and Resende, 1995)) are especially suitable and designed for discrete optimization problems, while others such as ACOR or Particle Swarm Optimization (PSO, Shi and Eberhart (1999)) are more convenient for continuous optimization problems.

Motivated by the success of ACO algorithms in discrete optimization problems, several algorithms that aim to extend ACO to continuous domains have been proposed, such as Continuous ACO (CACO, (Bilchev and Parmee, 1995)), Continuous Interacting Ant Colony (CIAC, (Dréo and Siarry, 2002)) and Ant Colony Optimization for Continuous Domains (ACOR, (Socha and Dorigo, 2006)). However, neither CACO or CIAC algorithms qualify to be an extension of ACO (Socha and Dorigo, 2006). On the one hand, these algorithms introduce mechanisms that do not exist in ACO metaheuristics (CACO adds the notion of nest and CIAC introduces direct communication between ants). On the other hand, neither of them considers incremental construction of solutions, which is one of the main characteristics of the ACO metaheuristics. On the contrary, as Socha and Dorigo claim, ACOR is the first extension of ACO metaheuristic to continuous domains that does not make any major conceptual change and maintains the main characteristics of the ant based algorithms (Socha and Dorigo, 2006).

The main differences between the discrete ACO based algorithms and ACOR are due to the domain of their decision variables. While the ants in discrete ACO based algorithms choose the values of the discrete decision variables c^n from a finite domain by sampling them from a probability mass function that uses the information stored in the pheromone table $\boldsymbol{\tau}$, in ACOR the ants choose the values of the continuous decision variables from an infinite domain by sampling them from a probability density function that uses an archive $\mathscr{A}$ of solutions. Therefore, the mechanism to save the information learned from previous iterations is different: due to the infinite

domain of the variables optimized by ACOR it is not longer possible to save the learned information in a finite pheromone table, instead ACOR uses a finite archive $\mathscr{A}$ of the best solutions found by the algorithm.

5.2.2. Solving MTS with ACOR

This section explains how the learned information is saved in ACOR archive and how it is used to sample new solutions. Besides, we propose the incorporation of specialized ants that enable the algorithm to consider problem specific information. Lastly, we present the proposed MTS heuristic for continuous domains that take into account the NFZ of the environment and explain how the heuristic is used for constructing the heuristic ant tours.

5.2.2.1. Archive of Solutions

Due to the infinite domain of the solution components it is not longer possible to represent the pheromones with a table. Instead, ACOR uses an archive of solutions represented in Figure 5.5, where each of the K best solutions found by the algorithm is placed in a different row of the archive sorted by their fitness value and where each column stores the value of a different decision variable.

$$\mathscr{A} = \begin{array}{c|cccccc|c} {}^{1}\mathbf{c} & {}^{1}c^{1} & {}^{1}c^{2} & \cdots & {}^{1}c^{n} & \cdots & {}^{1}c^{L} & f({}^{1}\mathbf{c}) \\ {}^{2}\mathbf{c} & {}^{2}c^{1} & {}^{2}c^{2} & \cdots & {}^{2}c^{l} & \cdots & {}^{2}c^{L} & f({}^{2}\mathbf{c}) \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ {}^{k}\mathbf{c} & {}^{k}c^{1} & {}^{k}c^{2} & \cdots & {}^{k}c^{l} & \cdots & {}^{k}c^{L} & f({}^{k}\mathbf{c}) \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ {}^{K}\mathbf{c} & {}^{K}c^{1} & {}^{K}c^{2} & \cdots & {}^{K}c^{l} & \cdots & {}^{K}c^{L} & f({}^{K}\mathbf{c}) \\ \hline & G^{1} & G^{2} & \cdots & G^{l} & \cdots & G^{L} & \end{array}$$

Figure 5.5 Archive of best solutions used in ACOR, sorted by their fitness rank $f({}^{1}\mathbf{c}) < ... < f({}^{k}\mathbf{c}) < ... < f({}^{K}\mathbf{c})$.

Each of the columns of the archive of solutions $\mathscr{A}$ is used by ACOR to define a Gaussian kernel G^{l} (weighted sum of several one-dimensional Gaussian functions)

and employed to sample each solution component. The Gaussian kernel associated to each column of the archive (dimension of the problem $l = 1, ..., L$) is given by:

$$G^l = \sum_{k=1}^{K} \frac{{}^k\omega}{\sum_{j=1}^{K} {}^j\omega} \mathcal{N}({}^k\mu^l, {}^k\sigma^l) \tag{5.13}$$

where the number of Gaussians of the kernel K is equal to the number of solutions of the archive, having each Gaussian function an associated weight ${}^k\omega / (\sum_{j=1}^{K} {}^j\omega)$ (where ${}^k\omega$ is given by Equation 5.14), a mean value ${}^k\mu^l$ (given by Equation 5.15), and a standard deviation ${}^k\sigma^l$ (given by Equation 5.16).

The weight of each Gaussian of the kernel depends on the solution rank k, the total number of Gaussian K (number of solutions in the archive) and the locality of the search process parameter q. As it can be seen from Equation 5.14, solutions with lower ranks k (better fitness) have associated bigger weights. Besides, lower values of the parameter q make ACOR strongly prefer best-ranked solution.

$${}^k\omega = \frac{1}{qK\sqrt{2\pi}} e^{(k-1)^2/(2q^2K^2)} \tag{5.14}$$

The mean of each Gaussian function ${}^k\mu^l$ is equal to the corresponding solution component ${}^kc^l$.

$${}^k\mu^l = {}^kc^l \tag{5.15}$$

Finally, the standard deviation depends on the average distance from the solution component ${}^kc^l$ to the solutions component of the rest of solution in the archive, and on the parameter ξ. Moreover, higher values of ξ produce a slower convergence of the algorithm.

$${}^k\sigma^l = \xi \sum_{e=1}^{K} \frac{|{}^ec^l - {}^kc^l|}{K-1} \tag{5.16}$$

The archive is updated with the K best solutions at each ACOR iteration and its information is used to construct the solutions of the next iteration. At the start of

the algorithm, the solution archive $\mathscr{A}$ is initialized with K solutions generated by uniformly randomly sampling the solution space.

5.2.2.2. Solution Construction from the Archive of Solutions

In ACOR the information saved in the archive of solutions $\mathscr{A}$ is used at each algorithm iteration to construct the solutions (ant tours) of the M ants of the population. Solutions are constructed sampling from the Gaussian kernels defined with the information contained in $\mathscr{A}$. The sampling process of a solution from the archive is divided in two steps.

Step 1. The first step consists in choosing one Gaussian function from the set of K Gaussians that form the Gaussian kernels G^l defined by Equation 5.13. The probability of choosing a Gaussian function is the weight associated to the k-th Gaussian of the Gaussian kernel:

$$^kP = \frac{^k\omega}{\sum_{k=1}^{K} {}^k\omega} \tag{5.17}$$

As the weights of each Gaussian are higher for solutions of lower (better) ranks according to Equation 5.14, the Gaussians associated to better solutions have higher probabilities of being chosen. The selected kernel index ($k \sim {}^kP$) is then used for sampling the values of the all the decision variables in the second step.

Step 2. Then, each solution component is sampled progressively from a Gaussian function with mean equal, according to Equation 5.15, to the l-th solution component of k-th solution from the archive selected at step 1 and standard deviation dependent, according to Equation 5.16, on the distance of $^kc^l$ to the other values of c^l within G^l.

This two-phase process makes the values of each variable $^kc^l$ within a solution $^k\mathbf{c}$ be sampled independently of each other and preferably around the values of the solutions of higher rank within $\mathscr{A}$, which stores the K best solutions obtained by ACOR up to each iteration. Although, the Gaussian function used for sampling each solution component at step 2 have different mean and standard deviation given by Equations 5.15 and 5.16, all the Gaussian functions used by an ant are associated

with a single solution ${}^{k}c$ chosen at step 1. This, as explained in (Socha and Dorigo, 2006), allows exploiting the correlation between variables.

5.2.2.3. ACOR with Heuristic Information

As stated before, unlike previous attempts to extend ACO techniques to continuous domains, ACOR maintains two intrinsic characteristics of ACO techniques for discrete optimization problems: incremental solution construction and indirect interchange of information between ants. However, contrary to the ACO algorithms in the discrete domain, it does not consider the possibility of including problem specific heuristic information during the solution construction process. Nevertheless, the inclusion of problem specific knowledge can be a great advantage in high complex problems like MTS. In fact, as we show in Chapter 4, the inclusion of MTS knowledge in MMAS allowed us to obtain higher quality results in less computational time. For this reason, we propose a modification of ACOR in order to let it include heuristic information (Pérez-Carabaza et al., 2017).

To achieve it, we consider a small group of ants that construct their solution (ant tour) using uniquely information of a problem specific heuristic. This heuristic ants are included in two different parts of ACOR: 1) in the initial archive created at the beginning of the algorithm and 2) at the solution construction of each iteration of ACOR. In both cases, only a percentage p_H of all the solutions are built considering heuristic ants (i.e. $p_H \cdot K$ ants of the solutions of the initial archive and $p_H \cdot M$ of the solutions built in each step of the algorithm).

It is worth noting that specialized ants have already been used in previous works. For instance (Madadgar and Afshar, 2008) considers explorer ants in order to introduce mutation operators. Besides, the ACO based algorithm ACE (Ant Colony Extendend) proposed by (Escario et al., 2015) starts with an empty pheromone table and considers two type of ants: patrollers (ants that use the pheromone table and heuristic information according to a decision policy) and foragers (ants that, unless the pheromone table is empty, only use pheromone information). Our heuristic ants

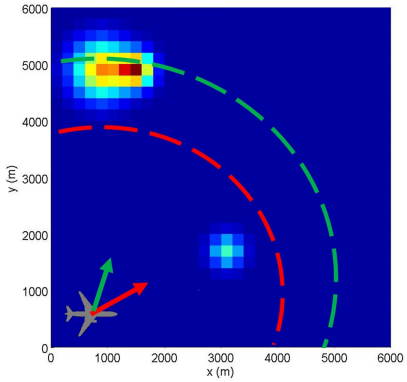
within ACOR follow a similar strategy to the patrollers of ACE, but they only exploit heuristic information.

The heuristic ants construct their solutions with the heading commands returned by the proposed MTS heuristic function, which uses the information about the current "unnormalized belief" state to guide the UAVs towards the highest probability areas as soon as possible. The proposed MTS function and the process followed to construct the whole tours of heuristic ants is described below.

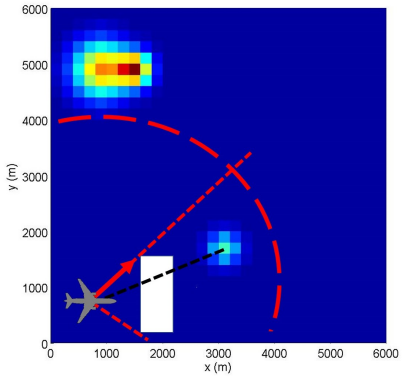
5.2.2.3.1. MTS Continuous Heuristic Function The proposed MTS heuristic uses the information about the current belief state (unnormalized probability map $\tilde{b}(v^t)$ and UAV state s_u^t) to guide the UAVs to the higher probability areas as soon as possible. This is done by returning the heading angle that points towards a relative maximum of the belief: the cell with maximum probability within a certain distance. Moreover, in order to provide a variety of possible heuristic solutions, the considered distance value by the u-th UAV (d_u) is chosen randomly for each solution, using a uniform distribution within a range from the scale (cell dimension) to the largest dimension that is reachable for the UAV within the planning horizon. In addition, as in MTS we want to reach maximum probability areas as soon as possible, the cells with the same probability values within the sampled distance are sorted taking into account their distance to current UAV state (the closest the best). Finally, in case there is a NFZ in the way to the destiny cell, the heading is modified to avoid it.

To better illustrate our MTS heuristic, we use the two examples displayed in Figure 5.6. On the one hand, in the scenario of Figure 5.6 (a) the colored arrows represent the headings that would be returned by the heuristic for two different distances. When $d_1 = 4200$ m, the MTS heuristic returns the heading represented with the green arrow that points to the global maximum of the belief within the green circle, while when $d_1 = 3300$ m, the heuristic guides the UAV, according to the red arrow, toward the smaller and closer probability area within the red circle. On the other hand, in the scenario of Figure 5.6 (b) there is a NFZ represented by the white rectangle over the probability map. In this case, the line toward the destiny cell (black dashed line) in-

tersects the NFZ and therefore the commanded heading is modified in order to make the UAV avoid it (red dashed line).



(a) For $d_1 = 3300$ m (red) and $d_1 = 4200$ m (green)



(b) For $d_1 = 3300$ m (red), modified heading due to the NFZ (white rectangle)

Figure 5.6 Schemes with the headings returned by the MTS heuristic (colored arrows) for a given distance radius (colored circular dashed lines).

Algorithm 9 Heuristic Ants Construction Process

function GENERATEWITHHEURISTIC($\tilde{b}(v^l), \mathbf{s}^l_{1:U}, l, L, \alpha$)
  for $k = 1 : \alpha$ **do** ▷ For each solution to be generated
    $\tilde{b}(v^t) = \tilde{b}(v^l)$ ▷ Initialize belief
    $s^t_{1:U} = s^l_{1:U}$ ▷ Initialize states
    $d_{1:U} \leftarrow$ **sampleDistances**(U)
    for $t = l+1 : l+L$ **do** ▷ For each time instant
      $\tilde{b}(v^t) \leftarrow \sum P(v^t|v^{t-1})\tilde{b}(v^{t-1})$ ▷ Predicted belief
      for $u = 1 : U$ **do** ▷ For each UAV
        ${}^k\theta^t_u \leftarrow$ **heuristicFun**$(\tilde{b}(v^t), s^{t-1}_u, d_u)$ ▷ MTS Heuristic
        $s^t_u = ObtainTrajectoryStep(s^{t-1}_u, \theta^t_u, \dot{s}_u = f(s^t_u, \theta^t_u))$ ▷ Simulate
        $\tilde{b}(v^t) \leftarrow (1 - P(D^t_u|s^t_u, v^t))\tilde{b}(v^t)$ ▷ Update belief
      end for
    end for
  end for
  return ${}^{1:\alpha}\theta^{l+1:l+L}_{1:U}$
end function

5.2.2.3.2. Solution Construction from Heuristic Information The process described in Algorithm 9 constructs the tours of α heuristic ants with the headings returned by the MTS heuristic function, that considers the updated belief $\tilde{b}(v^t)$ and the current state of the UAVs s_u^t. To construct the solution of each ant (main loop from line 2 to 14), first the "unnormalized belief" and UAV states are initialized with the updated "unnormalized belief" and current UAV locations (lines 3 and 4), which in case of the first step of the multi-stepped approach (when $q = 1$) respectively coincide with the $\tilde{b}(v^0)$ defined by Equation 5.9 and with the initial UAV locations $s_{1:U}^0$. Then, the algorithm samples a distance for each UAV ($d_{1:U}$) from a uniform distribution ranging from the scale of a cell to the maximum with of the search area (line 5). Next, for each time step of the solution construction loop (line 6 to 13), first the algorithm updates the "unnormalized" belief with the target dynamic model (line 7) and then, within a loop for each UAV (from line 8 to 12) obtains the next commanded heading angle from the MTS heuristic (line 9, Section 5.2.2.3.1), passes it through the UAV dynamical model (line 10) and updates the belief with the sensor model (line 11). At the end, Algorithm 9 returns the commanded heading sequences ${}^{1:\alpha}\theta_{1:U}^{l:l+L-1}$ of the α ants.

5.2.2.4. MTS Algorithm based on ACOR

Algorithm 10 shows the pseudocode of the proposed MTS algorithm based on ACOR. The algorithm optimizes a sequence of commanded headings $\theta_{1:U}^{c,l+1:l+L}$ for a given initial belief $\tilde{b}(v^l)$ and UAV states $s_{1:U}^l$ and is prepared to be called from the multi-step algorithm defined in Algorithm 8. Besides, we extend our previous notation using bold for the variables corresponding to a whole solution of the population and lower indexes on their right side for indicating the individual/ant (e.g. $\boldsymbol{s}_m$ is the solution $s_{1:U}^{l:l+L}$ the m-th ant among the existing M solutions and $\boldsymbol{\theta}_m$ its corresponding sequence of commanded headings $\theta_{1:U}^{c,l+1:l+L}$). It is worth noting that, when the first step is being optimized ($l = 0$); $s_{1:U}^l$ coincides with the initial UAV locations $s_{1:U}^0$ and $\tilde{b}(v^l)$ is equal to $\tilde{b}(v^0)$ defined by Equation 5.9.

Apart from the updated "unnormalized belief" and UAV states the algorithm also requires the initial time step l, the planner horizon of the subsequence L and the fitness vector FOC. In addition, it requires the percentage of heuristic ants p_H and ACOR general parameters: convergence speed ζ, locality of the search process q and archive size K. The algorithm uses the following functions:

- *GenerateWithHeuristic* function constructs α new solutions (ant tours) using problem specific heuristic information. The construction process (described by Algorithm 9) and the proposed heuristic (that guides the UAVs toward higher probability areas) was detailed in Section 5.2.2.3. This function is used in two different parts of the algorithm. First, during the initialization of the archive $\alpha = p_H \cdot K$ heuristic ants are created (in line 3) and then, at every algorithm iteration $\alpha = p_H \cdot M$ heuristic ants form part of the new population (in line 13).

- *GenerateRandomly* function constructs ants tours using a uniform distribution between the permitted heading commands. It is used in 4 line to construct the remaining $(1 - p_H)$ percentage of ants of the initial archive.

- *ObtainTrajectories* function obtains the UAV search trajectories according to the UAV dynamic model described in Section 5.1.1.1 and a sequence of commanded headings.

- *Evaluate* function evaluates a solution according to the criteria described in Section 5.1.2.2 and returns their fitness values saved in the fitness vector FOC^{l+L}. The pseudocode of the evaluation function is displayed in Algorithm 11. From lines 2 to 5, the feasibility objectives ($FOC[NFZ]^{l+L}$ and $FOC[COL]^{l+L}$) are computed with Equations 5.5 and 5.6, adding up their values in the current optimization step with the ones obtained in previous optimization steps ($FOC[NFZ]^{l}$ and $FOC[COL]^{l}$). Next, within the loop from lines 7 to 10, the term of the expected time corresponding to current solution ($s_{1:U}^{l:l+L}$) is computed, and added to $FOC[ET]^{l}$ in order to obtain $FOC[ET]^{l+L}$. Finally, in line 11 the myopia heuristic reduction criterion is obtained with Equation 5.10 (considering the last UAVs positions $s_{1:U}^{l+L}$) and saved in $FOC[MYOP]^{l+L}$.

- *UpdateArchive* function sorts jointly the solutions of the archive $\mathscr{A}$ and the new population of solutions according with the values stored in FOC (corresponding to the evaluation fitness criteria described in Section 5.1.2.2). Then, the operator updates $\mathscr{A}$ with the K best solutions and returns the new archive and its associated fitness $FOC_{\mathscr{A}}$ (line 12). It is worth noting that in line 10, as the initial archive is still empty, *UpdateArchive* function simply sorts the K solutions used to initialize the archive.

  Besides, in order to sort the solutions within the archive, we have to decide how to make ACOR (Socha and Dorigo, 2006), a mono-objective optimization approach, sort the solutions of $\mathscr{A}$ according to the values of the multiple criteria (constraints and evaluation objectives) obtained by *Evaluate*. To deal with the constraints we follow a commonly constraint handling technique (Deb, 2000) and sort the solutions in ascending order of the constraint violation function $\#_{\text{NFZ}} + \#_{\text{COL}}$, prioritizing feasible solutions ($\#_{\text{NFZ}} + \#_{\text{COL}} = 0$) to unfeasible ones ($\#_{\text{NFZ}} + \#_{\text{COL}} > 0$). Then, solutions with the same constraint violation values are sorted according to their performance criteria regarding the Pareto definition following the multi-objective approach for ACOR proposed in (Garcia-Najera and Bullinaria, 2007), which is in fact analogous to the survivor selection method of NSGA-II (Deb et al., 2002). Therefore, solutions are sorted according to 1) their feasibility and 2) the Pareto fronts of the performance criteria (and in case that it is necessary to select a part of the last Pareto front that fit in $\mathscr{A}$, solutions are chosen according to their crowding distance (Deb et al., 2002)).

- *GenerateWithArchive* function samples new solutions from the solutions stored in the archive of solutions $\mathscr{A}$. The sampling process is described in Section 5.2.2.2.

- *SelectBest* function chooses the best solution ${}^{gb}\boldsymbol{s}$ from the first front of the archive of solutions $\mathscr{A}$, which in case of mono-objective optimization coincides with its first (best ranked) solution.

Algorithm 10 MTS ACOR

1: **function** ACOR($\tilde{b}(\nu^l), s^l_{1:U}, l, L, FOC^l$)
Require: ξ, q, M, K, p_H ▷ ACOR parameters
2: $\alpha = p_H \cdot K$ ▷ Number of initial heuristic solutions in $\mathscr{A}$
3: $\boldsymbol{\theta}_{1:\alpha} \leftarrow GenerateWithHeuristic(\tilde{b}(\nu^l), \mathbf{s}^l_{1:U}, l, L, \alpha)$ ▷ Section 5.2.2.3, Alg.9
4: $\boldsymbol{\theta}_{\alpha+1:K} \leftarrow GenerateRandomly(U, L, K-\alpha)$ ▷ Initialize non-heuristic ants in $\mathscr{A}$
5: **for** k=1:K **do** ▷ For each solution of the archive
6: $s^{l:l+L}_{1:U} \leftarrow ObtainTrajectories(s^l_{1:U}, \boldsymbol{\theta}_k, \dot{s}_u = f(s^t_u, c^t_u))$
7: $FOC^{l+L} \leftarrow Evaluate(\tilde{b}(\nu^l), s^{l:l+L}_{1:U}, l, L, FOC^l, P(\nu^t|\nu^{t-1}), P(\overline{D}^t|\nu^t, s^t_u))$
8: $\boldsymbol{s}_k \leftarrow s^{l:l+1}_{1:U}, FOC^{l+L}_k \leftarrow FOC^{l+L}$ ▷ Store current information
9: **end for**
10: $[\mathscr{A}, FOC_{\mathscr{A}}] \leftarrow UpdateArchive([], [], \boldsymbol{\theta}_{1:K}, FOC^{l+L}_{1:K})$ ▷ Initialize and sort $\mathscr{A}$
11: $\alpha = p_H \cdot M$ ▷ Number of heuristic solutions of the population
12: **while Stop** $\neq true$ **do**
13: $\boldsymbol{\theta}_{1:\alpha} \leftarrow GenerateWithHeuristic(\tilde{b}(\nu^l), \mathbf{s}^l_{1:U}, l, L, \alpha)$ ▷ Section 5.2.2.3
14: $\boldsymbol{\theta}_{\alpha+1:M} \leftarrow GenerateWithArchive(\mathscr{A}, M-\alpha)$ ▷ Section 5.2.2.2
15: **for** m=1:M **do** ▷ For each ant of the population
16: $s^{l:l+L}_{1:U} \leftarrow ObtainTrajectories(s^l_{1:U}, \boldsymbol{\theta}_m, \dot{s}_u = f(s^t_u, c^t_u))$
17: $FOC^{l+L} \leftarrow Evaluate(\tilde{b}(\nu^l), s^{l:l+L}_{1:U}, l, L, FOC^l, P(\nu^t|\nu^{t-1}), P(\overline{D}^t|\nu^t, s^t_u))$
18: $\boldsymbol{s}_m \leftarrow s^{l:l+1}_{1:U}, FOC^{l+L}_m \leftarrow FOC^{l+L}$ ▷ Store current information
19: **end for**
20: $[\mathscr{A}, FOC_{\mathscr{A}}] \leftarrow UpdateArchive(\mathscr{A}, FOC_{\mathscr{A}}, \boldsymbol{\theta}_{1:M}, FOC^{l+L}_{1:M})$ ▷ Update $\mathscr{A}$
21: $[{}^{gb}\boldsymbol{s}, {}^{gb}FOC^{l+L}] \leftarrow SelectBest(\mathscr{A}, FOC_{\mathscr{A}})$ ▷ Select best solution
22: **end while**
23: **return** ${}^{gb}\boldsymbol{s}, {}^{gb}FOC^{l+L}$
24: **end function**

Summarizing, the algorithm starts with the generation of $p_H \cdot K$ of heuristic ants and $(1-p_H)K$ non-heuristic ants whose values are uniformly randomly generated. Then, those solutions are iteratively simulated according to the UAV dynamic model, evaluated, stored, and used (in line 10) to initialize the archive $\mathscr{A}$. Next, within the main loop (lines 12 to 21) a percentage of the population is formed with new heuristic ants (line 13) and the rest of ants are sampled from $\mathscr{A}$ (line 14). Then, within the loop from lines 15 to 19, the M ants of the population are iterativaly simulated, evaluated and saved. And at the end of the iteration, the combination of solutions of the population ($\boldsymbol{\theta}_{1:M}$) and of the archive $\mathscr{A}$ is sorted according to the

Algorithm 11 Evaluate Solution

1: **function** EVALUATE($\tilde{b}(\nu^l), s_{1:U}^{l:l+L}, l, L, FOC^l, P(\nu^t|\nu^{t-1}), P(\overline{D}^t|\nu^t, s_u^t)$)
2: $FOC[COL]^{l+L} \leftarrow FOC[COL]^l +$ ▷ Eq. 5.5
3: $\sum_{j=l}^{l+J}\sum_{u=1}^{U}\sum_{k=1+u}^{U} Collision(x_u^j, y_u^j, h_u^j, x_k^j, y_k^j, h_k^j)$
4: $FOC[NFZ]^{l+L} \leftarrow FOC[NFZ]^l +$ ▷ Eq. 5.6
5: $\sum_{j=l}^{l+J}\sum_{u=1}^{U} WithinNFZ(x_u^j, y_u^j) + CloseNFZ(x_{1:U}^J, y_{1:U}^J, \theta_{1:U}^J)$
6: **for** t=l+1:l+L **do** ▷ For each time step of the solution
7: $\tilde{b}(\nu^t) = \sum_{\nu^t \in G_\Omega} \prod_{u=1:U} (1 - P(D_u^t|\nu^t, s_u^t)) \sum_{\nu^{t-1} \in G_\Omega} P(\nu^t|\nu^{t-1})\tilde{b}(\nu^{t-1})$ ▷ Eq. 5.8
8: $FOC[ET]^t \leftarrow FOC[ET]^t + \sum_{\nu^t \in G_\Omega} \tilde{b}(\nu^t)$ ▷ Eq. 5.7
9: **end for**
10: $FOC[MYOP]^{l+L} \leftarrow \sum_{\nu^{l+L} \in G_\Omega} \prod_{u=1}^{U} H(\nu^{l+L}, s_u^{l+L})\tilde{b}(\nu^{l+L})$ ▷ Eq. 5.11
11: **return** FOC^{l+L}
12: **end function**

fitness criteria and the K best solutions are used to update $\mathscr{A}$ (line 20). Finally, once the stop condition is reached the global best found solution ${}^{gb}\boldsymbol{s}$ is returned.

Finally, in order to put our ACOR algorithm within the loop of the multi-step algorithm presented in Algorithm 8, we need to select a unique solution $s_{1:U}^{l:l+L}$ at the end of each optimization step (whose final UAVs states will define the initial ones of the next optimization step). To this end, we add *SelectBest* operator at the end of ACOR algorithm, which chooses a unique final trajectory ${}^{gb}\boldsymbol{s}$ among the ones that are presented in the best Pareto front identified by ACOR. To do it, it rounds the indexes of all the performance criteria except the last one with a predefined resolution and sorts the solutions within the first Pareto front according to their rounded FOC and a preselected priority ordering. In the results presented in this thesis, we prioritize *MYOP* criterion over *ET* and use two decimals of precision for MYOP performance criterion. With this strategy, the proposed *ACOR* algorithm returns the solutions with: the lowest rounded MYOP and the lowest ET value among the solutions with the lowest rounded MYOP. The myopia heuristic reduction criterion is preferred to ET because MYOP is favored by trajectories that 1) have already passed by regions where $\tilde{b}(\nu^t) \neq 0$ and that 2) finalize in good locations for the future control actions,

while ET is only favored by 1). Finally, the resolution is selected after multiple experiments over different scenarios and is related to the differences in the order of magnitude of the criteria.

It is worth noting that this evaluation strategy was followed in the MTS algorithms proposed in (Pérez-Carabaza et al., 2016b) and (Pérez-Carabaza et al., 2017). More concretely, in the GA based MTS algorithm presented in (Pérez-Carabaza et al., 2016b) we complement the performance criteria with the fuel consumption (which only affects to windy scenarios) and a criterion that prefers smooth solutions by minimizing the abrupt changes of the commanded heading signal. Besides, in the constrained MTS ACOR algorithm proposed in (Pérez-Carabaza et al., 2017) we use the described approach for the optimization of a unique performance criteria (ET), because in that case (or as it happens in the one-step approach followed in this work) it is not necessary a myopia heuristic reduction criterion.

5.3. Results

In this section we analyze the performance of the proposed MTS approach based on ACOR. To this end, we do a statistical analysis of the performance of the proposed approach over several search scenarios. In the first part of the analysis we examine the performance of the ACOR based MTS algorithm, focusing on determining the power of the MTS heuristic used by the heuristic a multi-stepped approach. Besides, we compare the results obtained by ACOR based MTS planner with the ones obtained by the GA based planner proposed by (Pérez-Carabaza et al., 2016b), which is the only approach in the literature review of probabilistic search algorithms presented in Chapter 2 that uses a complex UAV dynamic model and that optimizes multiple criteria.

5.3.1. Scenarios Setup

In order to test the performance of the proposed algorithm we have selected several search scenarios with different initial probability maps, number of UAVs, NFZ and target dynamics, which have been previously used to test the performance of GA and ACOR based planners in (Pérez-Carabaza et al., 2016b) and (Pérez-Carabaza et al., 2017). The characteristics of each scenario are explained below and represented in Figure 5.7 (which shows, over the scenario of each figure, if the target is static or dynamic, the number of UAVs U and planning horizon N). Moreover, the initial probability maps $P(\nu^0)$ are represented with colored/height matrices, where warmer colors and higher heights indicate areas with higher probability of target presence. The UAV initial states $s^0_{1:U}$ are represented with gray arrows and the UAV non-detection measurements with lines from the UAV positions to the ground. Besides, the cells that belong to a Non-Flying Zone (NFZ) are colored in white. Finally, the target dynamics $P(\nu^t|\nu^{t-1})$ of the non-static targets are sketched with orange arrows (over a top view of the belief in this case for a better visualization).

- *Scenario A* has the belief concentrated in three different zones of Ω, one with higher probability on the east and two lower probability areas on the west. The search is carried out by a unique UAV that should overfly the three high probability areas.
- *Scenario B* has two separated high probability zones. Thus, the two searching UAVs should cooperate to overfly both areas within the limited planning horizon while avoiding the NFZ situated in the center of Ω.
- *Scenario C* is used to see the performance of the MTS algorithms over more complex scenarios with multiple NFZ. The two UAVs have to avoid several NFZ while collecting as much probability as possible during a limited planning horizon.

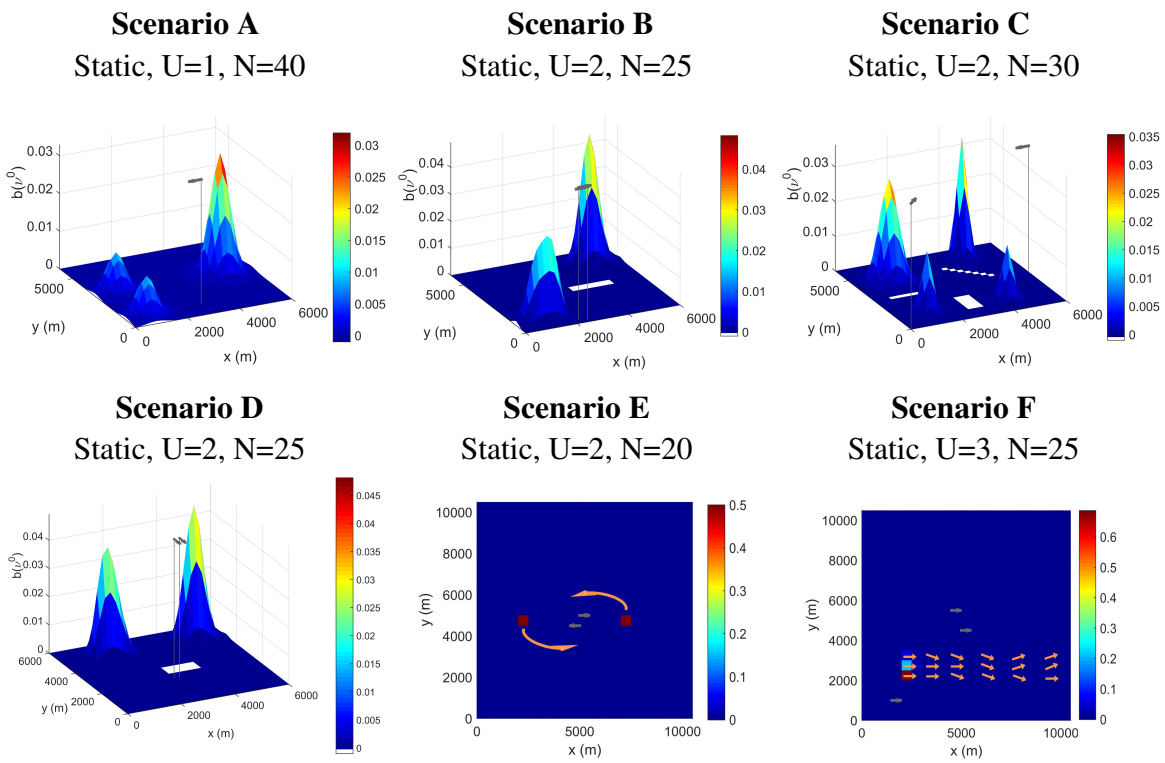


Figure 5.7 Search scenarios, initial probability maps represented with colored matrix, UAV initial states with grey arrows, NFZ with white cells and target dynamics with orange arrows.

- *Scenario D* has a NFZ in the center of the search area that should be surrounded by the two UAVs in order to reach the two high probability areas situated in the north of Ω.
- *Scenario E* has the belief initially concentrated in two high probability areas that, as time passes, are spread while making a circular motion. The two UAVs should cooperate to follow the belief movements avoiding collisions between them.
- *Scenario F* has a target dynamical model that simulates the movements of a lost boat in the sea. The three UAVs must gather as much belief as soon as possible without colliding.

5.3.2. Comparison Methodology

Due to the stochastic nature of the considered algorithms it is required to do a statistical analysis of their performance. Following a similar comparison strategy to Chapter 4, we have run each algorithm NR=20 times over each scenario and analyzed the fitness criteria of the best solution found at each algorithm iteration through the use of the dominance and mean ET evolution graphs.

As detailed in Section 5.1.2.2, in order to evaluate the solutions we consider the collision and NFZ avoidance criteria as feasibility constraints and the expected time and myopia heuristic reduction criterion as performance criteria. Despite the optimization of multiple criteria, we focus the analysis on the ET, as it is the most important objective for MTS. On one hand, the purpose of the myopia heuristic reduction criterion is to allow obtaining good final ET values when the optimization is divided in several optimization steps. On the other hand, as feasible solutions are always considered better then non feasible ones, we take into account the feasibility of solutions in the analysis and graphs, but we focus on the main MTS objective; the expected time of target detection.

Furthermore, in order to be able to compare the results obtained during the iterations of intermediate steps of the multi-stepped approaches with single-stepped variants we consider the ET expressions given by Equation 5.18. In particular, as $ET(s_{1:U}^{1:N})$ is not available until the last section of the trajectory is being optimized, for the computation times where the multi-stepped algorithms have not yet started the final section optimization, we approximate the value of $ET(s_{1:U}^{1:N})$ with the following expression, which accounts for the ET up to the section that is being optimized plus the worst possible case (associated to the case in which the UAVs do not collect more belief after $s_{1:U}^{qL}$) for the remaining time steps.

$$ET(s_{1:U}^{1:N}) = \sum_{l=0}^{qL} \sum_{v^l \in G_\Omega} \tilde{b}(v^l) + \sum_{l=qL+1}^{N} \sum_{v^{qL} \in G_\Omega} \tilde{b}(v^{qL}) \tag{5.18}$$

The two types of comparative graphs used in the analysis are described below.

Dominance evolution graphs show the dominance relationship between the different approaches along the iterations of the algorithms. The dominance graphs of the different algorithms/variants analyzed are shown in the first and third rows of Figures 5.9, 5.11, 5.13, 5.15 and 5.16. In order to determine if the results are statistically different, we apply the Wilcoxon test (with a significance level of 5%) to compare the results obtained by the algorithm selected as base with the results obtained by the other algorithms at the mean computation time of each iteration. More concretely, the comparison is performed with the results obtained at the mean computation time of each iteration of the base algorithm against the results obtained by the iteration with the closest (equal or small) mean computation time of the other algorithms. As only the ET values of feasible solutions are directly comparable, we compare the ET values of the different approaches if all the found solutions by the different runs of each algorithm are feasible, and distinguish if an approach has not find feasible solutions in all the simulations. We represent the outcomes of the comparisons in the dominance evolution graphs, displaying the comparison results of each algorithm/variant in different rows (whose labels indicate the algorithm under analysis) at the different computation times of the base algorithm (x-axis) using the following colors: green if the approach indicated in the y-axis dominates the base algorithm, red if the base algorithm dominates the approach indicated in the y-axis, gray if there is not statistical difference, black when the method in the y-axis has still not finished when the first iteration of the base algorithm has ended and blue if any of the solutions found by the approach indicated in the y-axis is not feasible. Therefore, the results of the iterations of the algorithms in green are statistically better than the results of the base algorithm, in gray similar and in red or blue statistically worse (unless the base algorithm is also blue, situation where a comparison of their feasibility values would be necessary to indicate the dominance relationship). Besides, the base algorithm is always represented in the first row of the graphs and indicated in the captions of the figures. Lastly, as the base algorithm can not dominate itself, the first row of this type of graphs is always gray.

ET evolution graphs show the evolution of the expected time (ET) along the algorithms iterations. To construct these types of graphs, we use the stored computation times and ET values of the best feasible solutions found at each algorithm iteration to calculate the mean computation time, mean ET and ET standard deviation at each algorithm iteration. This information is presented in the ET evolution graphs that appear in the second and fourth rows corresponding to each scenario of Figures 5.9, 5.11, 5.13, 5.15 and 5.16. The ET evolution graphs represent with different colors for each algorithm/variant its mean ET (colored line) and standard deviation (delimited by a shadowed area) against their computation time. Besides, we mark with dots over the mean ET curve the mean computation times of each algorithm/variant iterations. And hence, the computation time required for an algorithm iteration can be estimated by the time difference of two consecutive dots of its ET evolution curve.

Besides, as described in Section 3.4.2, it is worth noting that in both graphs we consider the expected time instant of target detection (the expected time when $\Delta T = 1$). To obtain the corresponding ET in time units we only have to scale the $ET(s_{1:U}^{0:N})$ obtained by Equation 5.7 by the considered measurement time lag ΔT.

Summarizing the information of both types of graphics complement each other. The dominance graphs show at the computation time of the base algorithm/variant if the algorithms have found feasible solutions and if so, if there is a statistical difference between the algorithms/variants under analysis but they do not show how different the solutions are. The ET evolution charts graphically represent the magnitude of the ET difference of the feasible solutions against the computation time, but they lack of the statistical significance information of the dominance evolution graphs.

5.3.3. Analysis of MTS-ACOR Performance

From now on we present the results conducted in order to analyze the performance of the proposed approach. First, we evaluate the ACOR based algorithm using a single step approach and focus the analysis of the power of the proposed constructive

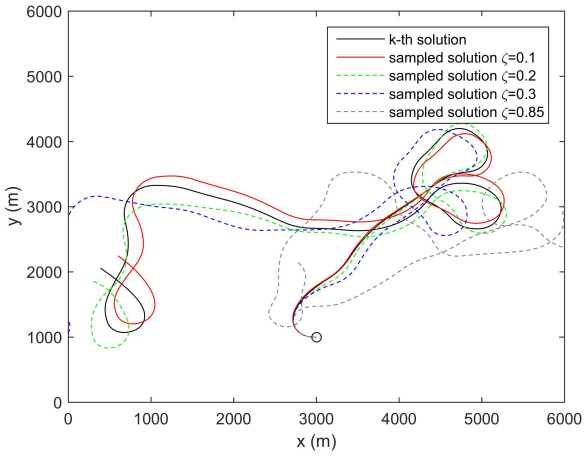


Figure 5.8 UAV search trajectories whose components were sampled from a Gaussian with mean equal to the solution components of the trajectory represented in black and standard deviations given by Equation 5.16 considering different ζ values indicated in the legend. The UAV initial position is indicated with a black circle.

MTS heuristic employed by heuristic ants. Then, we analyze the results obtained employing the multi-stepped planner and focus on the analysis of the capability of the proposed myopia heuristic reduction criterion. Finally, we compare the best algorithms from both ACOR base approaches with the GA based algorithm presented in (Pérez-Carabaza et al., 2016b).

5.3.3.1. Configuration of ACOR based MTS Algorithm

Before using a metaheuristic technique like ACOR we need to fix an appropriate set of parameters. This section describes ACOR parameters and lists the set of values used for the rest of the chapter.

ACOR characteristic parameters are: the archive size K, the locality of the search process q, the speed of convergence ζ and number of ants M. The archive size K determines the number of solutions (ants tours) that are saved in the archive of best solutions $\mathscr{A}$ and corresponds to the notion of population size in evolutionary algorithms (Socha and Dorigo, 2006). The locality of the search process or selection pressure parameter q influences in the weight ${}^{k}\omega$ associated to each solution of the archive, which are computed with Equation 5.14. For small values of q the best-

ranked solutions are strongly preferred (if q approaches zero only the best solution found so far is used for sampling solutions from $\mathscr{A}$) and for bigger values of q the weights of the different solutions of $\mathscr{A}$ become more similar. The parameter ζ influences in the standard deviation of the Gaussian kernels ${}^k\sigma^l$ given by Equation 5.16 and has an effect similar to that of the pheromone evaporation rate in ACO algorithms for discrete optimization. In order to explain how this parameter affects the sampling process, Figure 5.8 represents with different colors several search trajectories of a unique UAV sampled from the Gaussian function with mean equal to each solution component c^l of the trajectory represented in black and standard deviation ${}^k\sigma^l$ given by Equation 5.16 considering the different ζ values indicated in the legend. As it can been seen, the sampled solutions with bigger values of ζ differ more from the originally trajectory (represented in black). Hence, higher values of ζ allow higher exploration at the expense of a slower convergence of the algorithm. Moreover, it is necessary to fix the number of ants M used at each iteration, from which we consider a percentage of heuristic ants determined by the parameter p_H.

The set of chosen parameters values are listed in Table 5.1. As the archive size may not be smaller than the number of dimensions of the problem being solved (Socha and Dorigo, 2006), that is $N \cdot U$ for MTS, we set an archive size of $K = 80$. Besides, we make $q = 0.1$ as in (Socha and Dorigo, 2006). Finally, we make $\zeta = 0.2$. and consider a percentage of 10 per cent of heuristic ants ($p_H = 0.1$) from a total of $M = 20$ ants. Furthermore, for the multi-stepped approaches we set a maximum computational time for each optimization step of 20 seconds and for the single step variants we set a maximum computational time limit big enough to allow the algorithms to converge[1].

5.3.3.2. Single-Stepped ACOR with MTS Heuristic Ants

In this section we test the performance of the proposed ACOR based algorithm over the six search scenarios, focusing on the power of the proposed MTS heuris-

[1] All algorithms are implemented in MATLAB and run over a 2.81 GHz Intel Core i7 with 8GB RAM PC with Windows 10. Besides, the operations used to evaluate the ET of the solutions are speed up using the MATLAB Parallel Toolbox.

Table 5.1 Summary of the ACOR parameters used for MTS.

Parameter	Symbol	Value
Archive size	K	80
Locality of the search process	q	0.1
Speed of convergence	ζ	0.2
Number of ants per iteration	M	20
Percentage of heuristic ants	p_H	0.1

tic used by the heuristic ants and described in Section 5.2.2.3.1. With this purpose in mind, we compare several single step (SS) variants with different percentages of heuristic ants (p_H) whose parametrizations are summarized in Table 5.2. These variants are: the proposed algorithm (labelled as SS+H_{ants}) which considers a percentage of heuristic ants by setting $p_H = 0.1$, a variant without heuristic ants (labelled as *SS*) by setting $p_H = 0$ and a variant that only considers heuristic ants (labelled as H_{ants}) by setting $p_H = 1$. It is worth noting that, as all the algorithms of Table 5.2 consider a single step optimization, the myopic reduction criterion does not influence the evaluation criteria and only the feasibility objectives and expected time are optimized. Besides, this section extends the analysis of the ACOR-based MTS approach presented in (Pérez-Carabaza et al., 2017) with the comparison of the performance of the heuristic by itself (H_{ants}).

Table 5.2 ACOR variants under analysis in Section 5.3.3.2.

Short label	Multi-Step	Myopia heuristic reduction criterion	Heuristic ants	p_H
SS+H_{ants}			✓	0.1
SS				0
H_{ants}			✓	1

Figure 5.9 displays the dominance and ET evolution graphs of the results obtained by the three variants of Table 5.2 over the six analyzed search scenarios. For each scenario we represent the dominance evolution graphs of all the variants in the top rows and respectively the ET evolution graphs in the rows underneath.

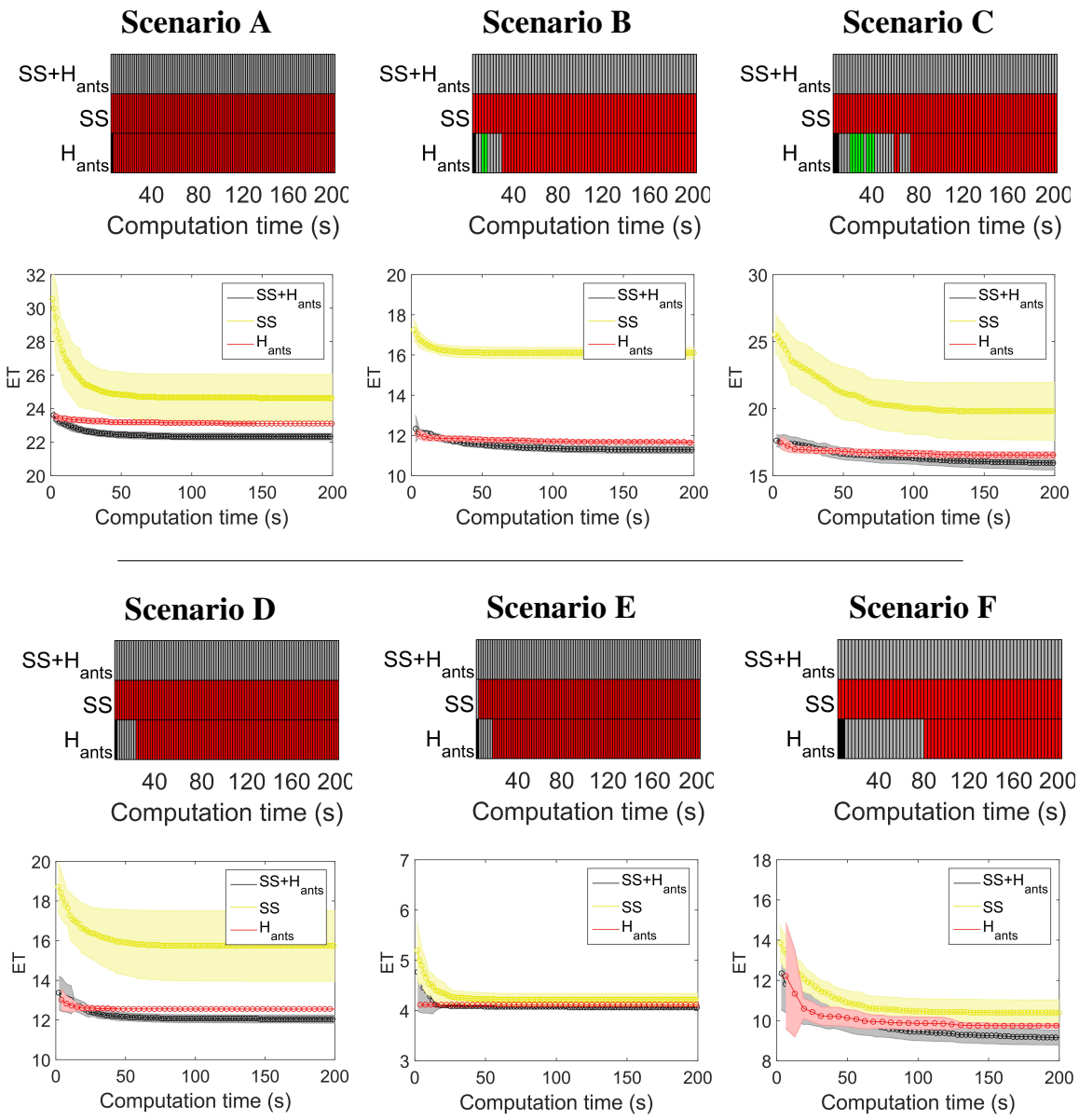


Figure 5.9 Comparison of the single-stepped ACOR based algorithms summarized in Table 5.2 over the six search scenarios. The selected base variant in dominance evolution graphs is SS+H_{ants}.

We have selected the proposed algorithm SS+H_{ants} as base algorithm (first row) for the dominance graphs. These graphs show that depending on the iteration and scenario some algorithms are dominated (red) or reach similar performance (gray) than the base algorithm, but all of them are able to find feasible solutions even from their first iteration (there is no blue).

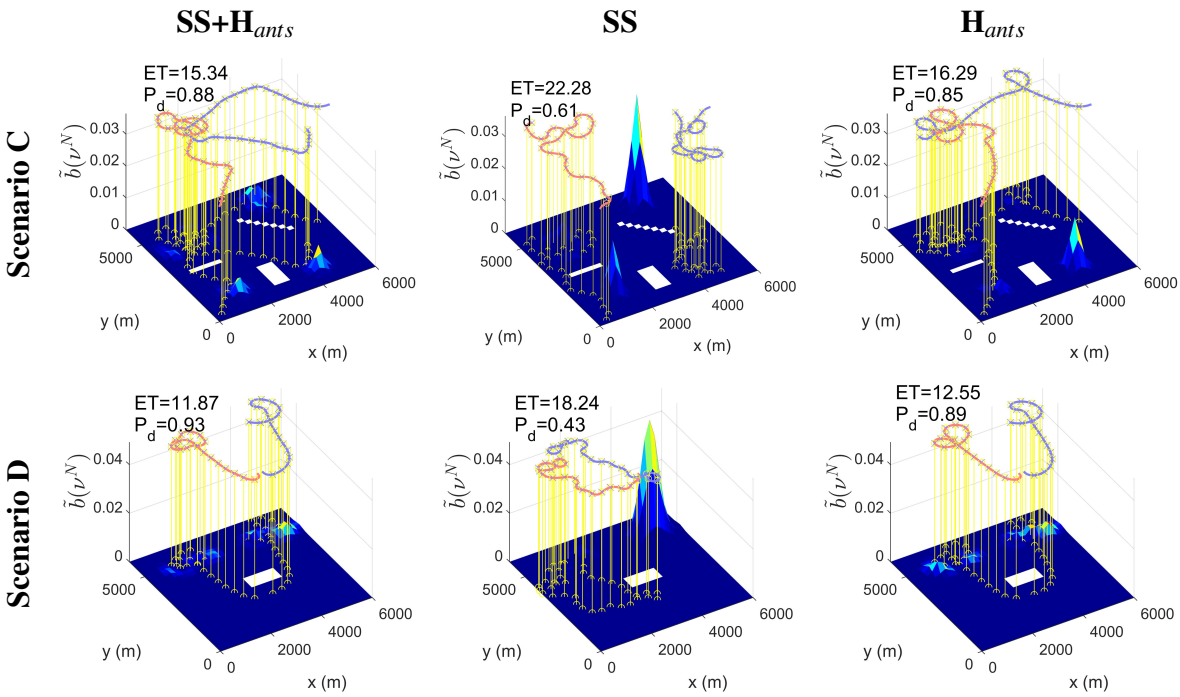


Figure 5.10 Representative solutions of the ACOR based single-stepped MTS algorithms indicated in the top labels for Scenarios C and D.

On the one hand, the dominance and ET evolution graphs show that the base algorithm (SS+H_{ants}) dominates the variant without heuristic information (*SS*) in all the scenarios even from the first iterations. Therefore, we can conclude that the proposed MTS heuristic is appropriate for MTS and its use allows our algorithm to quickly obtain higher quality solutions. This quality improvement is especially notorious in Scenarios B, C and D, as the ET evolution graphs of the second and forth rows of Figure 5.9 show.

On the other hand, the graphs of Figure 5.9 show that SS+H_{ants} obtains better results than the variant that only considers heuristic ants (H_{ants}). Although the heuristic by itself is able to obtain good quality solutions, the inclusion of non heuristic ants sampled from $\mathscr{A}$ (mechanism equivalent to the pheromones in the discrete ACO algorithms) allows the proposed MTS algorithm to reach higher quality solutions in all the scenarios.

Figure 5.10 shows a representative solution obtained by each of the variants analyzed for Scenarios C and D. In the case of Scenario C, the proposed algorithm is able to avoid all the NFZ and gather a high percentage of the initial probability ($P_d(s_{1:U}^{1:N}) = 0.88$). When no heuristic is used, *SS* variant is able to find feasible solutions that avoid the three NFZ but the proposed solution has considerable bigger expected time and lower probability of target detection. Finally, the solution proposed by H_{ants} is also feasible and although the UAVs are able to overfly some of the high probability areas, the solutions returned by SS+H_{ants} usually have lower ET and higher P_d values than the one proposed by H_{ants}. With regard to Scenario D, the solution returned by *SS* is clearly the worst. And although the solutions proposed by SS+H_{ants} is similar to the one returned by H_{ants}, the slightly closer circle done by the UAV in red allows to gather a 4% more of the initial belief (from $P_d(s_{1:U}^{1:N}) = 0.89$ to $P_d(s_{1:U}^{1:N}) = 0.93$). Besides, it is worth mentioning that the solution proposed by SS+H_{ants} was obtained by the modification of the solution proposed by H_{ants} (obtained by an heuristic ant) when sampled from the archive, in a similar way as it was shown in Figure 5.8. And thus, the use of non-heuristic ants helps the proposed algorithm to improve the quality of the trajectories obtained by the heuristic ants.

5.3.3.3. Multi-Stepped ACOR with Myopia Heuristic Reduction Criterion

In this section we analyze the proposed MTS algorithm based on ACOR focusing the study on the power of the proposed myopia heuristic reduction criterion (optimized following the evaluation approach described in Section 5.1.2.2) when solving the problems from the multi-stepped approach described in Section 5.1.2.3.

With the objective of analysing the adequacy of the myopia heuristic reduction criterion we compare the performance of the ACOR based multi-stepped algorithm (labelled as MS$_{Hm}$) with a version without the myopia reduction criterion (labelled as MS) over the six considered scenarios. For both multi-stepped algorithms we have divided the planning horizon N in $Q = N/L$ subsequences of a smaller planning horizon of $L = 5$ and considered a maximum computing time of 20 seconds for the optimization of each optimization step. Moreover, in order to test the advantage of

dividing the optimization problem in several subproblems or steps, we also analyze the single step ACOR variant (labelled as SS). Table 5.3 summarizes the characteristics of the three variants analyzed in this section. Note that with the purpose on focusing on the myopia heuristic reduction criterion none of them include heuristic ants ($p_H = 0$). Besides, all the variants optimize the feasibility objectives and the expected time, but only MS_{Hm} considers the myopia heuristic reduction criterion.

Table 5.3 ACOR variants under analysis in Section 5.3.3.3.

Short label	Multi-Step	Myopia heuristic reduction criterion	Heuristic ants	p_H
MS_{Hm}	✓	✓		0
MS	✓			0
SS				0

Figure 5.11 contains the ET dominance and evolution graphs of the analyzed variants over the six search scenarios. It is worth noting that, the plateaus of the ET evolution graphs and the worse performance during the first and intermediate iterations of the dominance graphs of MS_{Hm} and MS are due to their multi-stepped approach. As previously explained, the final solutions returned by all variants have a planning horizon N and thus, are directly comparable. However, in other to build the comparative graphs during the previous optimization steps (and compare trajectories of different lengths) we suppose the worst possible case (non overflying any region where $\tilde{b}(v^t) \neq 0$) for the remaining steps of the multi-stepped variants, Equation 5.18.

The dominance graphs show that in all the cases the algorithms under analysis found feasible solutions (i.e. search trajectories that avoid the NFZ and maintain the security distance between the UAVs) even for their first iterations.

On the one hand, if we compare the performance of MS with its single step variant SS we observe that both algorithms obtain similar performance in Scenarios B, D, E and F but SS outperforms MS in Scenarios A and C. Therefore, although optimizing the complete trajectories of horizon N at once with *SS* requires long time to converge in some scenarios (e.g. more than 100 seconds in Scenario C), due to its bigger

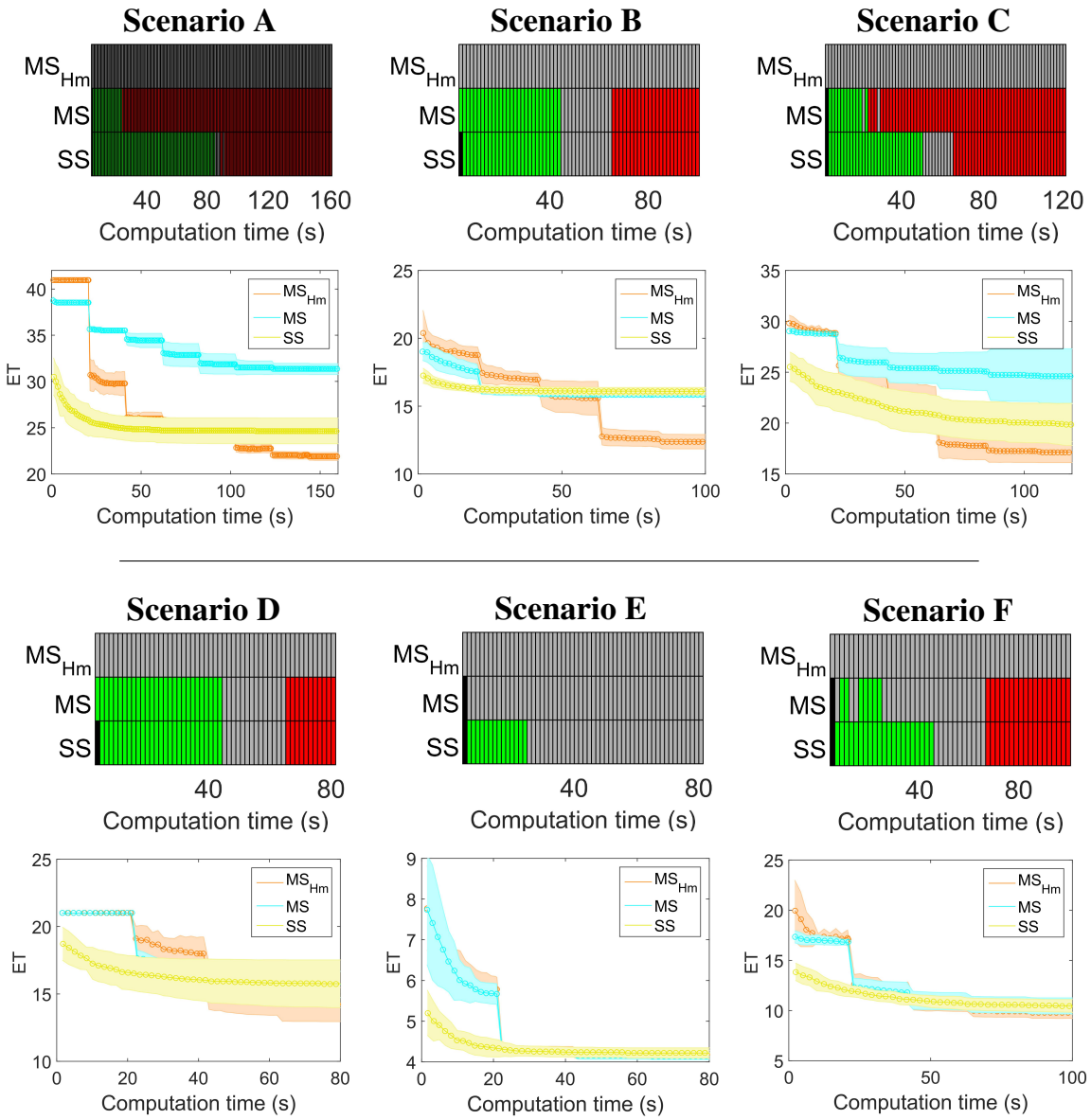


Figure 5.11 Comparison of the ACOR based algorithms resumed in Table 5.3 over the six search scenarios. The selected base variant in dominance evolution graphs is MS_{Hm}.

number of decision variables to optimize, the algorithm finally reaches similar or better performance than its multi-stepped variant. With regard MS, as it consider a small planning horizon at each optimization step, a computation time of 20 seconds is normally enough for the convergence of the optimization of each subsequence.

On the other hand, if we compare the performance of the multi-stepped algorithms with (MS_{Hm}) and without (MS) myopia heuristic reduction criterion we observe that

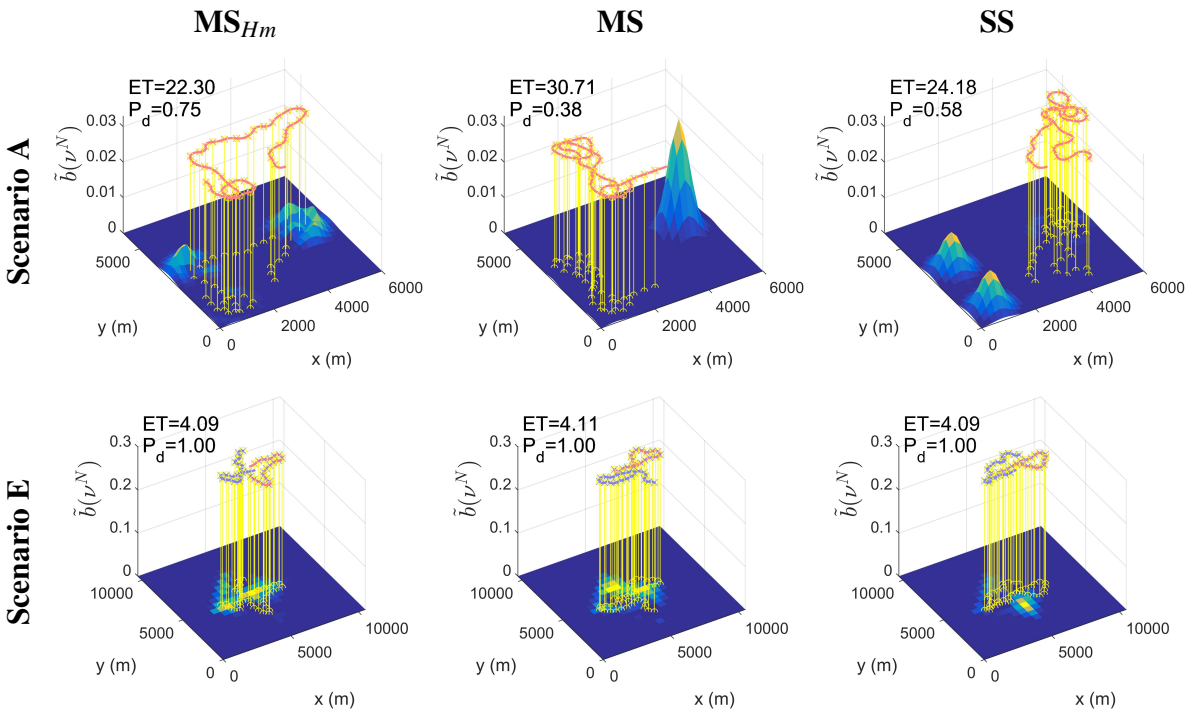


Figure 5.12 Representative solutions of the ACOR based MTS algorithms indicated in the top labels for Scenarios A and E.

MS$_{Hm}$ (selected as base variant in the dominance graphs) outperforms MS in the majority of the scenarios. As the ET evolution graphs show, this improvement is especially notorious in Scenarios A, B, C and D because, as the high probability areas are spread over the search area or far away from the UAVs initial positions, the myopic effects are bigger than in Scenarios D and F (where the higher values of $\tilde{b}(v^t)$ are close to the UAVs initial locations). We can conclude that the optimization of the proposed myopia heuristic reduction criterion helps to reduce the myopia of the final solutions, obtaining UAVs search trajectory with lower expected times by the estimation of the adequacy of the final UAVs positions of each optimization step for the remaining ones.

Figure 5.12 contains a representative solution of each of the algorithms under analysis for Scenarios A and E. In the case of Scenario A, on one side, as the higher probability area is further than the planning horizon L considered by MS, in the solution returned by MS the UAV only overflies the closer and smaller high probability

areas. On the other side, although all the probability areas are within the planning horizon N considered by SS, the complexity of this scenario is too high for optimizing the search trajectory at once and, in the solution proposed by SS, the UAV only overflies the highest probability area. On the contrary, the consideration of the myopia heuristic reduction criterion allows MS_{Hm} to obtain the solution with the best expected time, where the UAV overflies all the high probability areas. With regard to Scenario E, all the solutions have similar quality. In this scenario the areas with high $\tilde{b}(v^t)$ are within the planning horizon L of the MS approach and hence it proposes non-myopic solutions that have a similar quality to the one proposed by MS_{Hm}. Besides, the complexity of the scenario is not excessive to optimize it considering a single-stepped approach, and *SS* is also able to obtain a high quality solution.

5.3.3.4. Single-Stepped and Multi-Stepped ACOR Approaches Analysis

In the previous two sections we have analyzed single step variants with/without the heuristic ants and multi-stepped variants with/without myopia heuristic reduction criterion, concluding that the independent use of heuristic ants and myopia heuristic reduction criterion benefits the algorithm performance. In this section we compare the performance of the best ACOR based algorithms analyzed so far with a multi step algorithm (labelled as MS_{Hm}+H_{ants}) that considers both heuristic ants and optimizes the myopia heuristic reduction criterion.

Table 5.4 contains all the ACOR based MTS algorithms considered in this thesis and highlights the ones analyzed in this section: the single step variant with heuristic ants (SS+H_{ants}), the multi-stepped variant with myopia heuristic reduction criterion (MS_{Hm}) and a multi-stepped variant that considers both heuristic ants and optimizes the myopia heuristic reduction criterion (MS_{Hm}+H_{ants}). Moreover, Figure 5.13 contains the dominance and ET evolution graphs of these algorithms for the six analyzed scenarios. The selected base algorithm for the dominance graphs is MS_{Hm}+H_{ants}.

On one side, the dominance graphs show that MS_{Hm}+H_{ants} dominates its variant without heuristic ants (MS_{Hm}) in all the scenarios. Hence, we can conclude that the

Table 5.4 All ACOR algorithms analyzed in this thesis, the highlighted ones present the best performance and are analyzed in Section 5.3.3.4.

Short label	Multi-Step	Myopia heuristic reduction criterion	Heuristic ants	p_H
SS+H$_{ants}$			✓	0.1
SS				0
H$_{ants}$			✓	1
MS$_{Hm}$	✓	✓		0
MS	✓			0
MS$_{Hm}$+H$_{ants}$	✓	✓	✓	0.1

inclusion of a percentage of heuristic ants clearly improves the performance of the multi-stepped algorithm with the myopia heuristic reduction criterion.

On the other side, the multi-stepped approach (MS$_{Hm}$+H$_{ants}$) outperforms the single step variant (SS+H$_{ants}$) in Scenario A and reaches similar performance in the other scenarios, presenting a higher variance in Scenarios B and D. Besides, it is worth noting that the better performance of SS+H$_{ants}$ during the iterations corresponding to the first optimization steps is due to the evaluation of the ET according to Equation 5.18, which approximates the value of $ET(s_{1:U}^{1:N})$ accounting for the ET up to the current section that is being optimized plus the worst possible case (associated to the case in which the UAVs do not collect more belief from the final position of the current optimization step). Therefore, the comparison among single and multi-stepped approaches is only fair during the last optimization step, that is, when both approaches optimize the full UAV search trajectories. Although MS$_{Hm}$+H$_{ants}$ presents a slightly better performance than SS+H$_{ants}$, due to the good performance of both algorithms, the selection among the two algorithms can be based on their advantages. On the one hand, thanks to the use of heuristic ants the single-step algorithm SS+H$_{ants}$ can be used with high complex scenarios and has the advantage of returning the complete search trajectory after a reasonable computing time. Hence, it is an interesting approach for missions where the complete search path is required before the search missions starts (offline planning). On the other hand, MS$_{Hm}$+H$_{ants}$ benefits from the heuristic ants and is able to deal with high complexity scenarios by

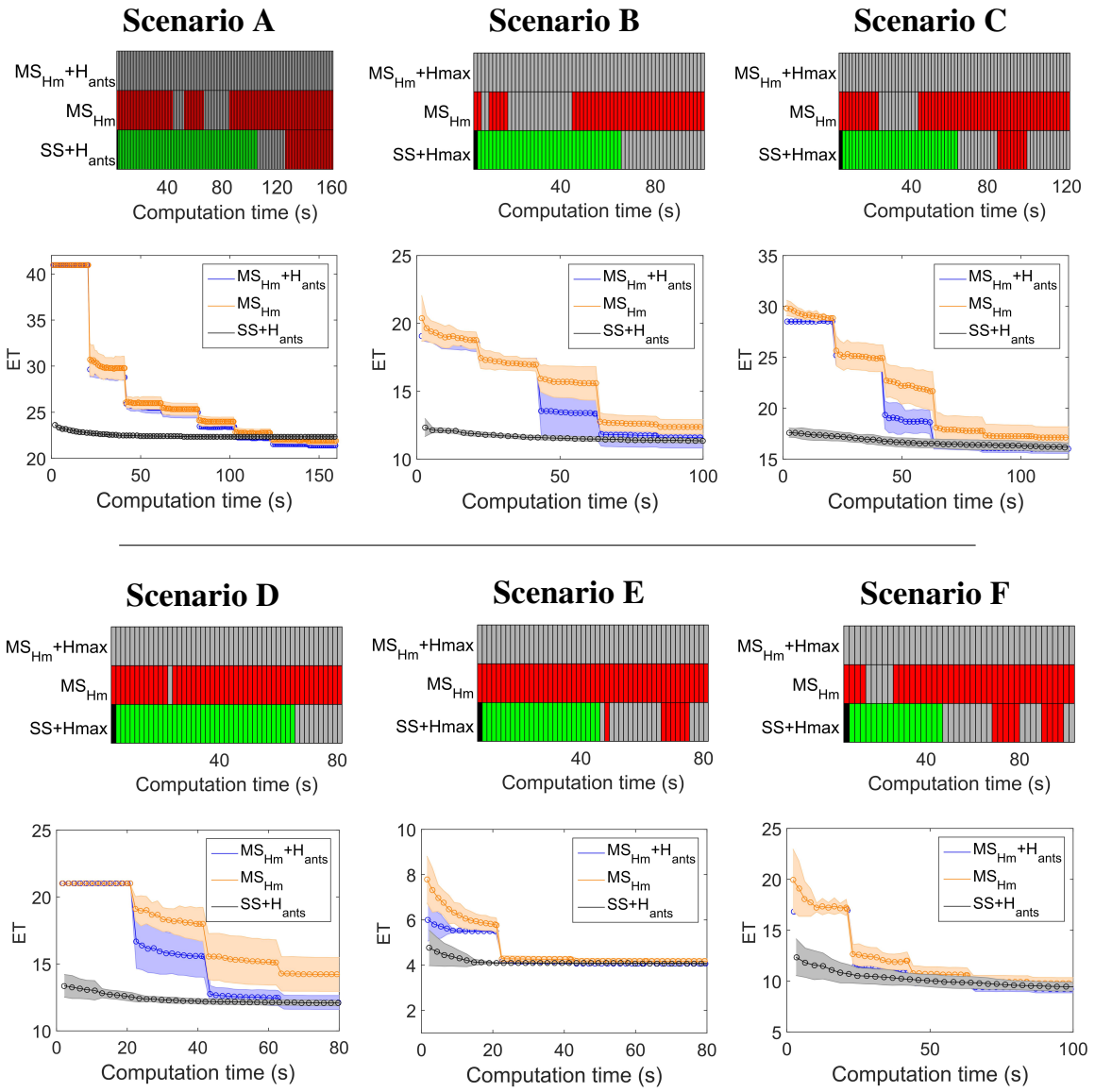


Figure 5.13 Comparison of the ACOR based algorithms resumed in Table 5.4 over the six search scenarios. The selected base variant in dominance evolution graphs is MS$_{Hm}$+H$_{ants}$.

means of a multi-stepped approach, returning non-myopic solutions thanks to the optimization of the myopia heuristic reduction criterion. The multi-stepped approach can be advantageous in online implementations for complex scenarios, where it is only necessary to compute the next trajectory step, which can be calculated during the flying time of the current trajectory step. In this way, the complexity of each optimization sub-problem can be controlled by the selection of an appropriate multi-stepped planning horizon (L).

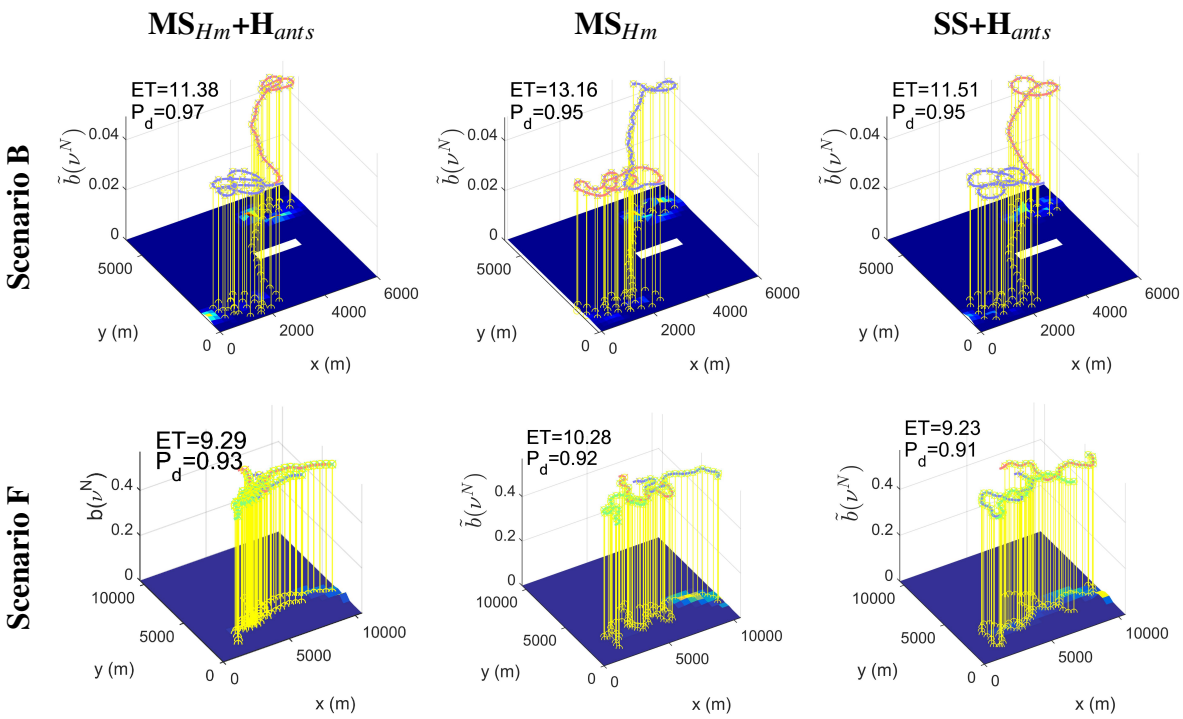


Figure 5.14 Representative solutions of the ACOR based MTS algorithms indicated in the top labels for Scenarios B and F.

Figure 5.14 shows a representative solution of each of the analyzed algorithms for Scenarios B and F. Regarding Scenario B, the three algorithms propose non-myopic solutions where the two UAVs distribute the search, flying one of them over the closer high probability area and the other one toward the further high probability area avoiding the NFZ. Besides, as the heuristic ants improve the solutions by directing more straightforward the UAV to the further high probability area, the search trajectories proposed by MS$_{Hm}$ are worse than the ones proposed by MS$_{Hm}$+H$_{ants}$ and SS+H$_{ants}$. With regard to Scenario F, the solutions proposed by the three algorithms gathered more than 90 per cent of the initial belief by means of following the belief movements described by $P(\nu^t|\nu^{t-1})$. Again, the two variants that consider a percentage of heuristic ants present better performance regarding the ET than MS$_{Hm}$.

5.3.4. Comparison with GA based MTS Algorithm

In this section we present a MTS algorithm based on Genetic Algorithms (GA), analyze its performance and compare it against the proposed ACOR based MTS algorithm.

5.3.4.1. Description of the GA based MTS Algorithm

Genetic algorithms are search metaheuristics inspired in the process of natural selection, which belong to the larger class of Evolutionary Algorithms (EA) that were first proposed by (Holland, 1992). GA are optimization algorithms whose iterative process can be divided in two steps. First, GA recombine and possibly randomly mutate the individuals of their population to generate new individuals. Then, GA stochastically selects the survivor individuals taking into account their fitness from the new and old candidate solutions (survival of the fittest). In this way, by using genetic operators, iteration by iteration, the population is evolved toward better solutions. One of the advantages of GA-based methods is their versatility, as they accept discrete and continuous design variables for the optimization process (in fact, a GA based MTS algorithm which optimizes discrete variables was already presented in Chapter 4).

In (Pérez-Carabaza et al., 2016b) we propose a multi-stepped MTS algorithm based on GA that performs a multi-objective optimization of the UAVs search trajectories codified as sequence of continuous optimization variables. Algorithm 12 presents the pseudocode of the GA-based MTS algorithm, which optimizes the UAVs search trajectories for a given initial "unnormalized belief" $\tilde{b}(\nu^l)$ and UAV states $s^l_{1:U}$. The algorithm can also be called from the multi-stepped receding horizon approach described in Algorithm 8, replacing the call for *ACOR* (Algorithm 10) by *GA* function (Algorithm 12). For the algorithm notation we use bold for the variables corresponding to a whole solution of the population and lower indexes on their right side for indicating the individual solutions (e.g. $\boldsymbol{s}_r$ is the r-th solution $s^{l:l+L}_{1:U}$ of a total of R solutions, and $\boldsymbol{\theta}_r$ its corresponding sequence of commanded headings $\theta^{c,l+1:l+L}_{1:U}$). At

the beginning of the algorithm, the *Initialize* function generates randomly (with a uniform distribution within [-180º,180º]) the values of the decision variables (sequence of UAV commanded headings) of the initial population of size R. And then (within the loop between lines 3 to 7) the initially population $\boldsymbol{\theta}_{1:R}$ is iteratively simulated by means of the UAV dynamic model and evaluated according with the evaluation procedure already described in Algorithm 11, saving in the fitness array FOC^{l+L} the feasibility and performance objective functions described in Section 5.1.2.2. Within the main loop, GA first selects the pairs of candidate solutions that the are then crossed (by *Crossover* function in line 10) and mutated (by *Mutate* function in line 11) to obtain a new offspring population (${}^{offs}\boldsymbol{\theta}_{1:R}$). Then the offspring population is iteratively simulated and evaluated (within the loop from line 12 to 16) and the new generation is selected (by *Survivors* function in line 17) from the offspring and previous population. This process continues until the stop condition is fulfilled and the best found solution ${}^{gb}\boldsymbol{s}$ with the best associated fitness ${}^{gb}FOC^{l+L}$ is returned as solution.

5.3.4.2. Comparative Results with GA based MTS Algorithm

In this Section we analyze the performance of the GA-based MTS algorithm over the six search scenarios under study and compare their results against the ones obtained by the MTS algorithm based on ACOR. More concretely, we first test the performance of the GA-based algorithm focusing on the power of the proposed myopia heuristic reduction criterion for avoiding falling within myopic solutions within the multi-step optimization. Then, we compare the performance of GA and ACOR based algorithms focusing on the analysis of the advantages and disadvantages of both approaches.

All the GA based variants analyzed within this section follow the general structure described in the previous section and consider the following operators and parametrizations: a population size of $R = 20$, a binary tournament selection operator (to avoid the converge of the algorithm to a premature suboptimal solution), a single point crossover operator with crossover probability ($p_{xover} = 0.8$) and a two-stepped addi-

Algorithm 12 MTS GA

function GA($\tilde{b}(\nu^l), s^l_{1:U}, l, L, FOC^l$)
Require: $P(\nu^0)$, $P(\nu^t|\nu^{t-1})$, $P(\overline{D}^t|\nu^t, s^t_u)$, $\dot{s}_u = f(s^t_u, c^t_u)$ ▷ Target and UAVs models
Require: R, p_{xover}, p_{mut} ▷ Population size and crossover and mutation probabilities
  $\boldsymbol{\theta}_{1:R} \leftarrow Initialize(U, L, R)$ ▷ Randomly generation of the initial population
  for r=1:R **do**
    $s^{l:l+L}_{1:U} \leftarrow ObtainTrajectories(s^l_{1:U}, \boldsymbol{\theta}_r, \dot{s}_u = f(s^t_u, c^t_u))$
    $FOC^{l+L} \leftarrow Evaluate(\tilde{b}(\nu^l), s^{l:l+L}_{1:U}, l, L, FOC^l, P(\nu^t|\nu^{t-1}), P(\overline{D}^t|\nu^t, s^t_u))$
    $\boldsymbol{s}_r \leftarrow s^{l:l+1}_{1:U}, FOC^{l+L}_r \leftarrow FOC^{l+L}$ ▷ Store current information
  end for
  while Stop $\neq true$ **do**
    ${}^{parent}\boldsymbol{\theta}_{1:R} \leftarrow Select(\boldsymbol{\theta}_{1:R}, FOC^{l+L}_{1:R})$ ▷ Select pairs of parents
    $\boldsymbol{\theta}_{1:R} \leftarrow Crossover({}^{parent}\boldsymbol{\theta}_{1:R}, p_{xover})$ ▷ Cross of the parents
    ${}^{offs}\boldsymbol{\theta}_{1:R} \leftarrow Mutate({}^{offs}\boldsymbol{\theta}, p_{mut})$ ▷ Mutate the offspring
    for r=1:R **do**
      $s^{l:l+L}_{1:U} \leftarrow ObtainTrajectories(s^l_{1:U}, {}^{offs}\boldsymbol{\theta}_r, \dot{s}_u = f(s^t_u, c^t_u))$
      $FOC^{l+L} \leftarrow Evaluate(\tilde{b}(\nu^l), s^{l:l+L}_{1:U}, l, L, FOC^l, P(\nu^t|\nu^{t-1}), P(\overline{D}^t|\nu^t, s^t_u))$
      ${}^{offs}\boldsymbol{s}_r \leftarrow s^{l:l+1}_{1:U}, {}^{offs}FOC^{l+L}_r \leftarrow FOC^{l+L}$ ▷ Store current information
    end for
    $[\boldsymbol{\theta}_{1:R}, \boldsymbol{s}_{1:R}, FOC^{l+L}_{1:R}] \leftarrow Survivors(\boldsymbol{\theta}_{1:R}, \boldsymbol{s}_{1:R}, FOC^{l+L}_{1:R}, {}^{offs}\boldsymbol{\theta}_{1:R}, {}^{offs}\boldsymbol{s}_{1:R}, {}^{offs}FOC^{l+L})$
    $[{}^{gb}\boldsymbol{s}, {}^{gb}FOC^{l+L}] \leftarrow SelectBest(\boldsymbol{\theta}_{1:R}, \boldsymbol{s}_{1:R}, FOC^{l+L}_{1:R})$
  end while
  return ${}^{gb}\boldsymbol{s}, {}^{gb}FOC^{l+L}$
end function

tive gaussian mutation (consisting in mutating all the values within ${}^{offs}\boldsymbol{\theta}_{1:R}$ with a gaussian noise of low variance $\sigma_l = 1$ and only a few values, selected uniformly with a low mutation probability $p_{mut} = 1/(L*U)$, with a gaussian noise of higher variance $\sigma_h = 100$). Besides, the survivors are selected according to NSGA-II (Deb et al., 2002) taking into account to the evaluation criteria described in Section 5.1.2.2. This parametrization was selected according to the GA-based MTS algorithm presented in (Pérez-Carabaza et al., 2016b), whose set up was fixed after a statistical analysis of the influence of different parametrizations over several scenarios. The consideration of the same evaluation criteria, solution codification and UAV models ensures a fair comparison between ACOR and GA approaches. Besides, the selection of the same

population size as in ACOR facilitates the comparison between the computational times required for each iteration of the algorithms.

5.3.4.2.1. Multi-Stepped GA with Myopia Heuristic Reduction Criterion In this section we analyze the GA based MTS algorithm performance focusing on the power of the proposed myopia heuristic reduction criterion. To do it, we follow a similar approach to the analysis of the MTS ACOR based algorithms presented in Section 5.3.3.3 and the GA based MTS algorithms presented in (Pérez-Carabaza et al., 2016b).

Table 5.3 summarizes the characteristics of the analyzed GA based MTS algorithms: the proposed GA algorithm (MS^{GA}_{Hm}), its variant without myopia heuristic reduction criterion (MS^{GA}) and with a single step approach (SS^{GA}). Moreover, Figure 5.15 contains the dominance and evolution graphs of the three variants over the analyzed scenarios, where the proposed GA based algorithm (MS^{GA}_{Hm}) is selected as base algorithm.

Table 5.5 GA variants under analysis in Section 5.3.4.2.1.

Short label	Multi-Step	Myopia heuristic reduction criterion
MS^{GA}_{Hm}	✓	✓
MS^{GA}	✓	
SS^{GA}		

With regard to the multi-stepped approach, if we compare the performance of the single step algorithm SS^{GA} with the multi-step MS^{GA} we can conclude that their performance depends on the scenario. On the one hand, due to the myopia effects MS^{GA} outperforms SS^{GA} in Scenario A and C but, on the other hand, in Scenario B and D the approach of dividing the optimization problem in several less complex subproblems is a better strategy.

Furthermore, in all the scenarios MS^{GA}_{Hm} outperforms (Scenarios A, B, C, D) or reach similar quality results (Scenario E and F) as its variant without myopia heuristic reduction criterion (MS^{GA}). Therefore, the graphs of Figure 5.15 clearly show the

benefit of including the proposed myopia heuristic reduction criterion in the multi-step optimization.

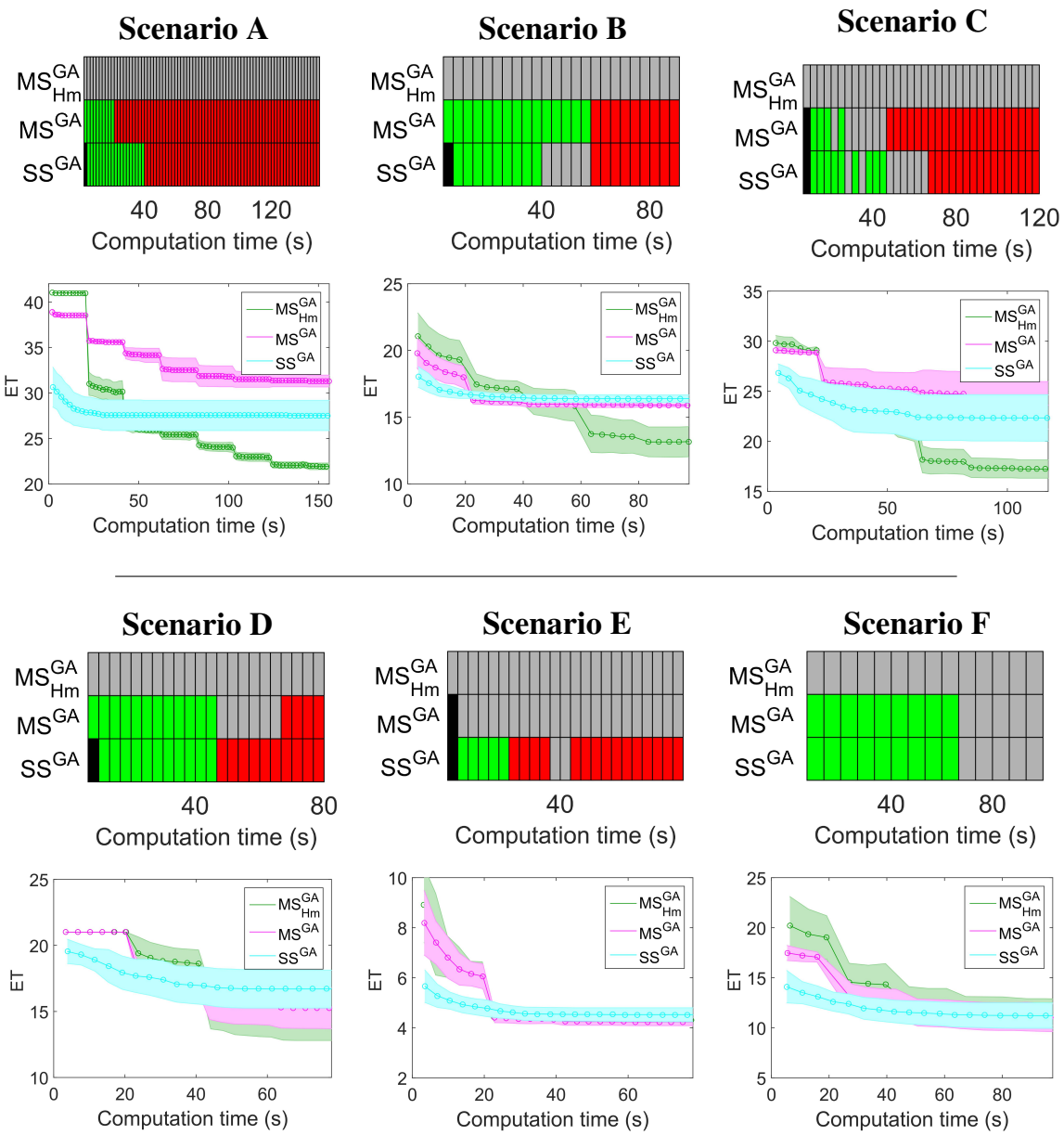


Figure 5.15 Comparison of the GA based algorithms resumed in Table 5.5 over the six search scenarios. The selected base variant in dominance evolution graphs is MS^{GA}_{Hm}.

5.3.4.2.2. Comparative Results of ACOR and GA based MTS Algorithm In this section we compare the performance of the two different kinds of MTS algo-

rithms proposed in this chapter, based on genetic algorithms (GA) and on the ant colony optimization algorithm for continuous domains (ACOR).

Summarizing, the main differences between both approaches are 1) the mechanism used to generate new solutions of each iteration (either by sampling the new individuals from the archive of solutions $\mathscr{A}$ in ACOR or through the modification or combination of the individuals of the population in GA) and 2) the consideration of using specific MTS information to construct a percentage of its solutions (within ACOR).

Table 5.6 summarizes the main characteristics of the algorithms analyzed in this section. We analyze the GA and ACOR based variants that presented the best performance (MS^{GA}_{Hm} and MS_{Hm}+H_{ants}) and in order to analyze the adequacy of GA or ACOR techniques by theirself, we include in the comparison the equivalent ACOR based algorithm of MS^{GA}_{Hm} (i.e. MS_{Hm}). Besides, the dominance and ET evolution graphs of the analyzed algorithms are presented in Figure 5.16. We have selected MS^{GA}_{Hm} as based algorithm for the dominance graphs, as this section is focused on its comparison with the ACOR based algorithms.

For the comparison of MS^{GA}_{Hm} and MS_{Hm} shown in Figure 5.16, we can conclude that the ACOR based approach (MS_{Hm}) reaches better (Scenarios B, E and F) or equal (Scenarios A, C and D) performance than the GA based approach (MS^{GA}_{Hm}). Therefore, the intrinsic mechanisms used by ACOR to generate new solutions (by the sampling process from $\mathscr{A}$) have better performance in some MTS scenarios than the mechanisms used by GA (based on genetic operators). However, when the ACOR based algorithm includes a percentage of heuristic ants, the proposed MTS algorithm (MS_{Hm}+H_{ants}) reaches better performance than MS^{GA}_{Hm} in all the scenarios.

To conclude, thanks to the optimization of the myopia heuristic reduction criterion both the GA and ACOR based algorithms (MS^{GA}_{Hm} and MS_{Hm}) reach good performance in all the scenarios. However, the inclusion of a percentage of heuristic ants enables the proposed ACOR based algorithm to outperform the GA based approach in all the scenarios.

Table 5.6 Comparison of the best ACOR and GA variants, analyzed in Section 5.3.4.2.2.

Short label	Multi-Step	Myopia heuristic reduction criterion	Heuristic ants	p_H
MS^{GA}_{Hm}	✓	✓	-	-
MS_{Hm}+H_{ants}	✓	✓	✓	0.1
MS_{Hm}	✓	✓		0

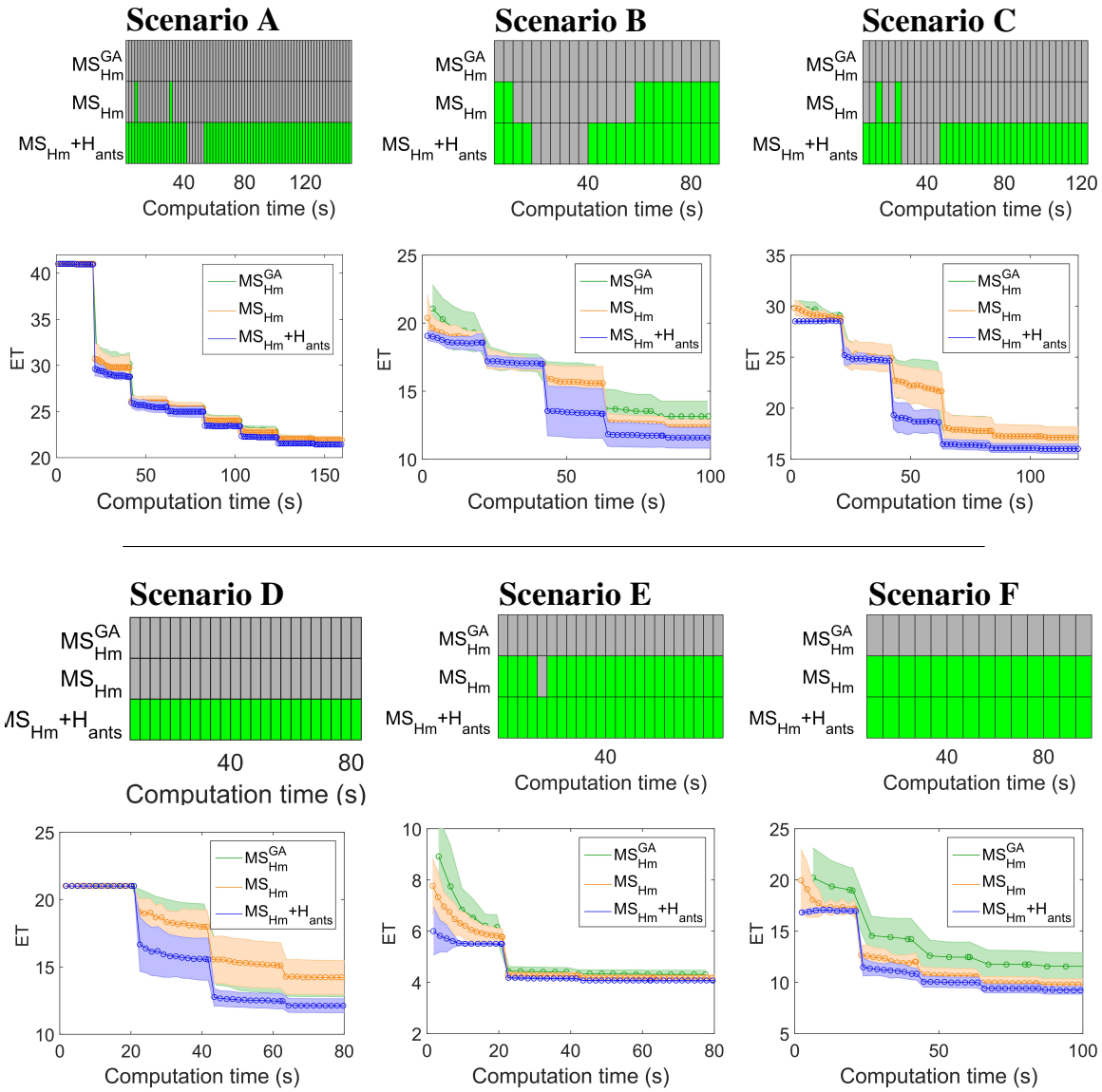


Figure 5.16 Comparison of the ACOR and GA based algorithms resumed in Table 5.3 over the six search scenarios. The selected base variant in dominance evolution graphs is MS^{GA}_{Hm}.

5.3.5. Summary

This chapter proposes a MTS algorithm based on the ant colony based algorithm for continuous optimization problems (ACOR) that optimizes the search trajectories of a fleet of UAVs looking for a target with uncertain location and dynamics.

The chapter commences describing the formulation of the problem, the codification of the solutions, the multi-objective evaluation criteria and the multi-stepped approach. The addressed MTS problem considers a radar model and a continuous UAV model implemented in Simulink, which can be parametrized according to the UAV characteristics such as maximum speed or bank angle. The proposed MTS algorithm optimizes both feasibility constraints (avoiding flying NFZ and collisions among the UAVs) and performance objectives (prioritizing solutions that fly first over the areas with higher probability of target presence). The use of complex dynamic motion and sensor models in the evaluation of the feasibility and optimization criteria permits the algorithm to obtain search trajectories that can be followed by fixed-wing UAVs. Besides, the proposed method enables to solve the optimization of the complete UAV search trajectories within a multi-stepped approach, by sequentially solving several optimization problems of smaller complexity (with smaller number decision variables to optimize).

Next, the chapter describes how the iterative ACOR algorithm generates the ant tours (by sampling them from an archive of the best found solutions) and how the MTS problem is formulated in order to optimize it using ACOR. Moreover, although the original ACOR does not use heuristic information, we consider, motivated by the benefits of the MTS heuristic shown in Chapter 4, a percentage of heuristic ants that construct their path using the information of a MTS heuristic that guides the UAVs toward the areas with higher probabilities of target presence.

Summarizing, we propose two strategies to deal with the problem complexity (intrinsic to MTS and increased with the use of the realistic UAV models and multi-objective evaluation criteria) whose benefits are proven over several search scenarios. On the one hand, the use of heuristic ants allows to obtain the complete search tra-

jectories in reasonable computational time and with better fitness than the algorithm variant without heuristic ants. On the other hand, the optimization of the proposed myopia heuristic reduction criterion within a multi-stepped approach allows the algorithm to sequentially obtain the search trajectories by solving less complex problems while avoiding the myopic effects derived from the multi-stepped approach. This approach can be advantageous for online implementations of the algorithm (where there are strict computational resources limitations and the next trajectory part can be optimized during the flight time of the previous one). Furthermore, these two strategies can be used in conjunction allowing to solve high complexity search scenarios in reasonable computational time.

Finally, the performance of the proposed ACOR based MTS algorithm is compared with a MTS algorithm based on GA, concluding that, thanks to the inclusion of a percentage of heuristic ants, the proposed ACOR based MTS algorithm reaches better results than GA in all the analyzed scenarios.

Chapter 6

MTS Planner Integration in Ground Control Station

"The science of today is the technology of tomorrow"

Edward Teller

The work presented in this thesis, funded by Airbus within SAVIER project, has been integrated and validated in two different Ground Control Stations (GCS). This chapter contains an overview of SAVIER project, a description of the MTS Planner designed for the validating purpose, a methodology to define the initial target probability map, and a general explanation of the integration process within the two GCS (Pérez-Carabaza et al., 2016a) and (Pérez-Carabaza et al., 2019).

6.1. SAVIER Project

The use of multiple UAVs and the development of new Ground Control Stations (GCS) computational capabilities are changing the typical GCS structure, where one operator is in charge of the unique UAV and other operator is responsible for the payload (sensors). The work presented in this thesis is part of a collaboration project between several Spanish universities and Airbus Group. In particular, SAVIER (Sit-

Figure 6.1 Sketch of the main research lines of SAVIER project.

uational Awareness VIrtual EnviRonment) project studies how to benefit from the application of different automatizations and artificial intelligence techniques in the next generation of GCS. The project has provided support to 12 theses whose research lines are described below and sketched in Figure 6.1:

- **Metaheuristics** applied to plan missions that encompasses several tasks (surveillance, fire extinction, etc) according to different objectives (Ramirez-Atencia et al., 2017), or, more specifically, to plan target search missions (this thesis).
- **Machine Learning** techniques to command and monitor the missions using the operator voice (de la Calle Silos, 2017) and hand gestures (Mantecón et al., 2014). Besides, the monitorization of the operators biological signals allows to determine their stress level (Hernando-Gallego and Artés-Rodríguez, 2015).
- **Data mining** to support operator training (Rodriguez-Fernandez et al., 2015), to keep the security of the new types of portable GCS (Paniagua Diez et al., 2015) and to monitor multi-UAV missions (Roldán et al., 2018).
- **Image processing** to support UAV air refueling tasks (Martín et al., 2016) and to benefit from augmented reality techniques during the payload monitoring (Ruano et al., 2017).

- **Advanced multimedia interfaces** to manage several UAVs with higher situational awareness (Ruiz et al., 2015).
- **Air Traffic Management Procedures** to analyze the protocols that enable multiple unmanned vehicles to share the search space (Cordón, 2017).

One of the main objectives of the project was to integrate the research contributions of all the theses in Airbus Research & Technology GCS, a simulator that allows to define and monitor ATLANTE[1] missions, and control and monitor its payload (onboard electro-optic sensor). Moreover, in order to show the added value of the project, a specific mission that incorporates the contributions of all the theses was designed and incrementally improved at each of the demonstrations performed every six months (after the initial year of the project). Besides, as Airbus Research & Technology GCS was desinged for working with an unique UAV, a multi-UAV GCS simulator based on QGroundControl was set up in order to test the contributions of the theses that deal with multiple UAVs: (Ramirez-Atencia et al., 2014; Roldán et al., 2015; Ruiz et al., 2015) and this thesis. Hence, in our case we have integrated our work in both GCS since, on the one hand, the integration within Airbus R&T GCS allows to test the technologies with a GCS whose design and communication protocols follows NATO[2] standards, and since on the other hand, the integration with QgroundControl GCS allow us to test our work with a multi-UAV GCS.

6.2. Software Architecture Design

The state of the art Probabilistic Search (PS) algorithms assume the target initial probability map as a given input and do not have any interface that acts as intermediary between the GCS operator and the algorithms. However, as part of SAVIER project we need to validate the proposed MTS algorithms over two different GCS. To this end, we have developed a MTS Planner with a Graphical User Interface (GUI)

[1] ATLANTE: Avión Táctico de Largo Alcance No Tripulado Español.
[2] North Atlantic Treaty Organization (NATO).

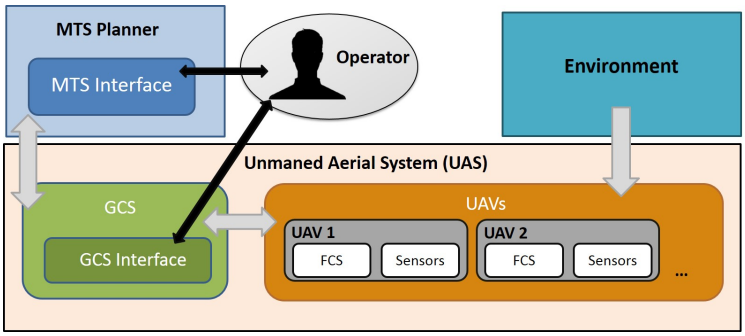


Figure 6.2 MTS Planner Integration scheme with a general GCS.

that allows the operator to define the search scenario, run the optimization, analyze the results and establish connection with the two GCS. To this end, we consider two different approaches. One possible architecture is to directly integrate the MTS Planner within each GCS. However, this will require to adapt the software for each GCS and develop a new planner GUI within each. Alternatively, we can develop an independent MTS Planner that acts as an auxiliary tool and allows to establish connection with any GCS. This option, selected for our MTS Planner, has the advantage of only requiring the design of a general MTS Planner instead of developing different ones for each GCS. In this way, the communication specifications differ depending on the GCS, but the MTS Planner is the same for both integration process.

Figure 6.2 contains a general scheme of the selected architecture, whose main elements are the MTS Planner, the Unmanned Aerial System (UAS) and the Environment. The UAS is composed by the GCS and the UAVs, having each UAV a Flight Control System (FCS) and onboard sensors. The GCS sends to the UAVs the commanded plan and the UAVs send back the information about their states and the environment information collected by their sensors. In case that the UAS is simulated (like the two cases considered in this thesis), the UAVs and environment are simulated, but the GCS should deal similarly with simulated or real identities. The MTS Planner and the GCS are communicated, the former sends the optimized search route and the later the initial search information and (once the mission has started) information about the mission state. The operator, in charge of monitoring and control the mission, can interact with the GCS and MTS Planner through their respective in-

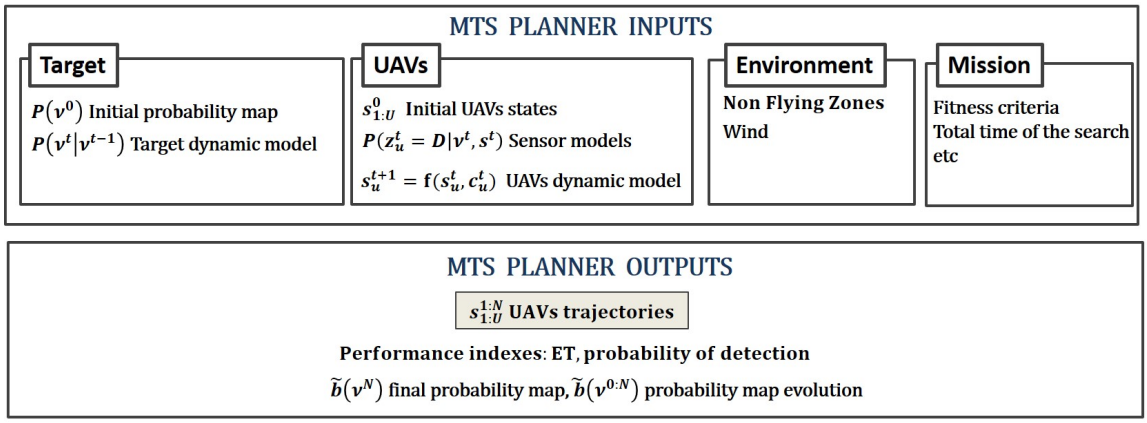


Figure 6.3 Main MTS Planner inputs and outputs.

terfaces. In this way, the MTS Planner acts like an auxiliary tool that allows operator to interact with it and send and listen information from the GCS.

6.3. MTS Planner

The main utilities of the designed MTS Planner are the following: to allow the operator/user to define the search scenario, to optimize the UAVs routes by means of the selected MTS algorithm, to analyze the results and to establish connection with two different GCS.

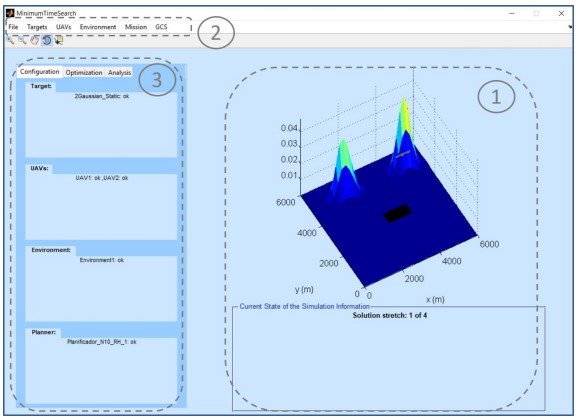
The basic inputs and outputs of the MTS Planner are schematized in Figure 6.3. The MTS Planner requires the MTS algorithm inputs: information/models relative to the target, UAVs, environment and general mission information (e.g. algorithm stop condition or parameters). Besides, the planner should facilitate the operator the definition, modification and visualization of all this information. As outputs, the MTS Planner should return the optimized UAV routes $s^{0:N}_{1:U}$, performance information (e.g. expected target detection time ET or probability of detection) and the evolution of the probability map $\tilde{b}(v^{0:N})$ corresponding to the optimized routes.

The MTS Planner should have a friendly interface that allows the operator to easily define the inputs, run the optimizations and analyze the outputs. In this regard, the following requirements were identified and considered for the MTS Planner design:

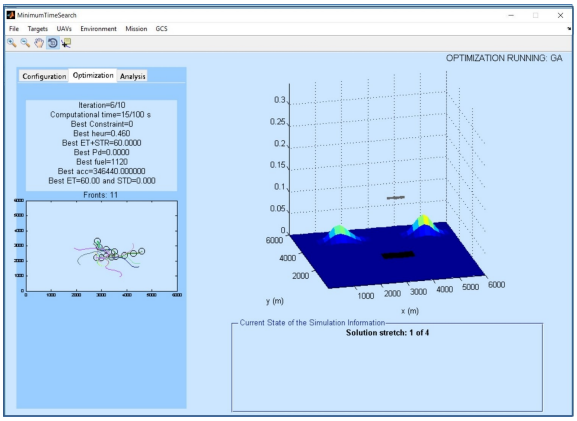
- The inputs and outputs should be clearly organized with the purpose of leading the operator through the scenario definition and simulation steps.
- There should be visual information associated to the majority of the inputs and outputs. For instance the probability map can be represented with the colored matrix.
- There should be visual information during the optimization process, that informs about the optimization algorithm evolution (e.g. current algorithm iteration or best found solution fitness values).
- The operator should be able to either use the MTS Planner when it is connected with a GCS (to optimize the UAVs routes for the search scenario information received from the GCS) or use the planner by itself (to optimize a scenario stored or defined using the planner).
- The operator should be able to store/load a completed scenario or to construct it element by element, defining the information relative to the target, UAVs, environment and mission.
- The operator should be able to store the results of an optimization and load them to reproduce them later.
- There should be contextual help support associated to the interactive elements of the simulator interface.

Taking into account the previous requirements, we have prototyped a MTS Planner whose GUI main window is displayed in Figure 6.4. More concretely, Figure 6.4 (a) contains a capture of the planner during the search scenario definition, Figure 6.4 (b) during the optimization process and Figure 6.4 (c) during the simulation of a solution. In the three stages, the information is divided in three different panels remarked with dashed grey lines in Figure 6.4 (a):

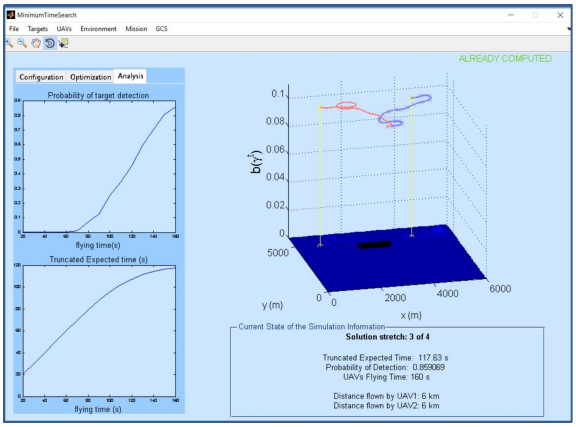
Panel 1. A general visualization panel allows the operator to quickly obtain a view of the scenario; the UAV initial states are indicated with gray arrows, the NFZ



(a) Search scenario definition



(b) Optimization



(c) Simulation of a solution

Figure 6.4 Main MTS Planner window during (a) scenario definition, (b) optimization and (c) simulation phases.

with black rectangles and the probability map is displayed over the search region with different colors and heights (where warmer colors and higher heights indicate higher probabilities of target presence). Besides, when a solution is being simulated this panel displays the UAV trajectories and the updated probability map, as shown in Figure 6.4 (c).

Panel 2. A menu with several emerging tabs allows to load, modify or display the information about the different inputs; the *Target* tab gives access to target belief and motion models, the *UAVs* tab to the UAVs state, motion model and sensor models,

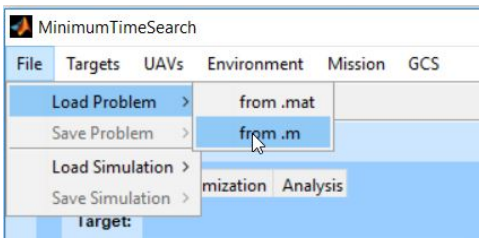


Figure 6.5 Loading a scenario from database through *File* menu tab.

the *Environment* tab includes different information relative to the environment and the *Mission* tab to the mission inputs (information concerning the optimization algorithm and fitness criteria). Besides, the *File* tab allows to load/save the scenarios or simulations, and the *GCS* tab to establish connection with the two GCS.

Panel 3. An informative panel that displays different content according to the selected tab, each one associated to each of the main phases of the MTS Planner: the information of initial phase of search scenario definition is presented in the *Configuration* tab, the information of the optimization phase in the *Optimization* tab, and the information of the simulation and solution analysis phase in the *Analysis* tab.

Besides, in order to build a search scenario the operator can either load a scenario between the ones previously saved in the database, listen to the search scenario information sent by one of the GCS (if a connection is available) or define separately each of the scenario elements. For example, Figure 6.5 shows how to load an scenario from the database with *File* menu tab.

Moreover, in order to define/modify/show the information of the different scenario elements, the operator can access the *Target, UAVs, Environment* and *Mission* tabs. For instance, as shown in Figure 6.6, through the *UAVs* tab we can edit or show information about the UAVs *State*, *Dynamical model* or *Payload* (i.e. onboard sensors). As an example, the window displayed in Figure 6.7 (accessed through *UAVs* $\rightarrow$ *Payload* $\rightarrow$ *Show* menu tabs) shows the information about the sensor models of each UAV. To this end, it allows to choose between three different views; the probability map when *Belief* button is selected (Figure 6.7 (a)), the sensor probability function when *Sensor* button is chosen (Figure 6.7 (b)), and the updated probability map with

a sensor measurement $z = \overline{D}$ when *Applied* button is pushed (Figure 6.7 (c)). In this way, this utility allows to see how the measurements affect the belief and hence is useful to adjust the sensor model.

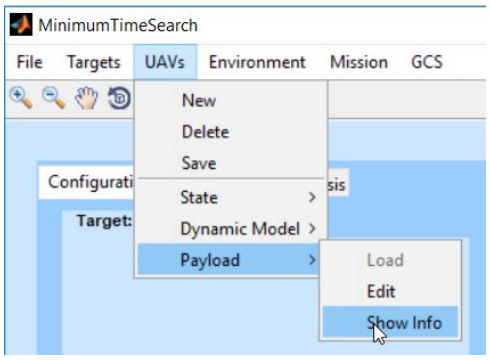
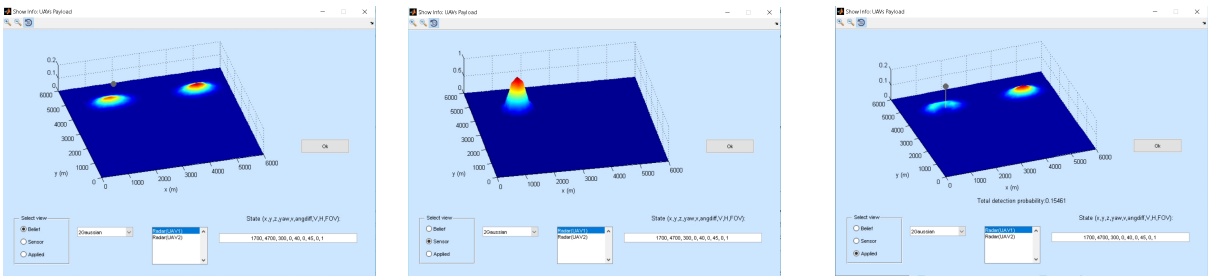


Figure 6.6 UAVs tab menu options.

Furthermore, the user can select the MTS algorithm from the implemented ones and modify their default parametrizations through the *Algorithm* tab (accessed through the *Mission* menu tab displayed in Figure 6.8). More concretely, as Figure 6.9 (a) shows, the MTS Planner allows to choose between GA or ACOR based MTS algorithms and to change their default parameters, which in the case of the GA based algorithm selected in Figure 6.9 (a) are the population size and the probabilities relative the crossover and mutation operators. Therefore, through the *Algorithm* tab it is possible change specific parameters of the optimization techniques. Alternatively, the *General* option of *Mission* menu tab (shown in Figure 6.8) allows to change general parameters (common to all the MTS algorithms). In particular, as Figure 6.9 (b)



(a) Belief view (b) Sensor model view (c) Applied measurement view

Figure 6.7 Window with information about the UAV sensor models.

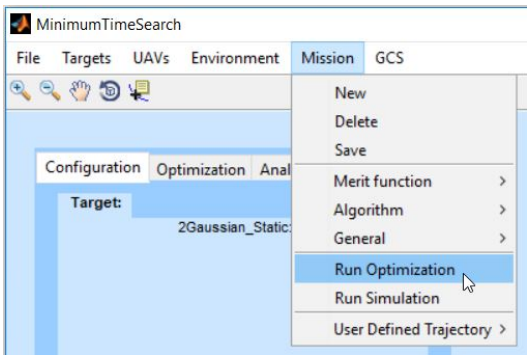
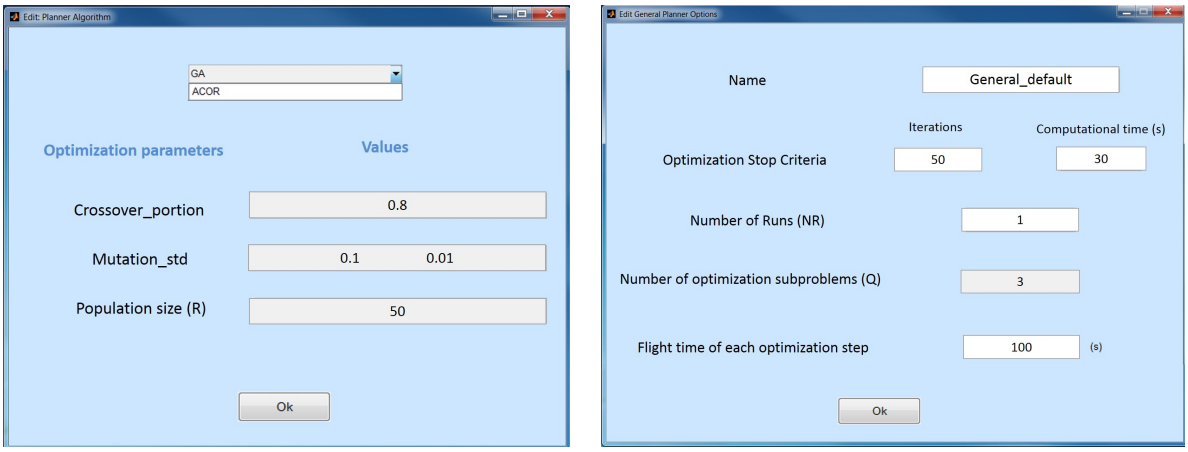


Figure 6.8 Mission tab menu options.

shows, the *General* option allows to change algorithm stop criteria (maximum iterations and/or computational time per optimization), the number of runs (NR), and the flight time of each optimization step (the flight time of the UAV trajectories optimized in a optimization step). Besides, this window also displays (without possibility of direct modification) the number of optimization steps (Q), which is automatically calculated by the coefficient of the maximum flight time specified for the mission (T) (introduced through the *UAVs* tab) and the flight time for each optimization step. Note that, with all this information and the time lag between control actions ΔT (defined trough *UAVs* menu tab), the planner can compute the planning horizon of the full trajectory (N) and the planning horizon of each subsequence ($L = N/Q$), which coincides with N in the single-step approaches (where $Q = 1$).

Once the scenario is defined, the user can run an optimization by selecting the option *Run Optimization* from the *Mission* menu tab, as shown in Figure 6.8. During the optimization phase, the MTS Planner displays information relative to the state of the optimization, such as the current algorithm iteration and fitness of the current best found solution (as shown in the example of Figure 6.4 (b)). And, as displayed in Figure 6.4 (c), once the optimization has finished, the MTS Planner simulates the proposed solution while displaying relevant information such as the expected target detection time, the target detection probability, etc.

To conclude, on the one hand, the MTS Planner tool helped to define the search scenarios presented in this thesis. On the other hand, it allowed to validate the pro-



(a) Optimization technique information (b) General MTS algorithms specifications

Figure 6.9 Windows accessed through the Mission menu tab enable to change (a) specific information about the optimization technique and (b) general parametrizations of the MTS agorithm.

posed MTS algorithms and technologies and to integrate them with two GCS (as described in Section 6.5).

6.4. Definition of the Target Initial Probability Map

As previously mentioned, within SAVIER project it was defined a mission that allows to show the theses contributions during the project demonstration. With the purpose of integrating the work developed in this thesis, we require to build an appropriate initial probability map for the mission and consequently, a procedure that allows the operator to define the initial belief in the MTS Planner was identified as a key requirement. This section describes the procedure followed for defining the initial probability maps used during the project demos.

Due to the lack of benchmark test data, the performance of PS algorithms is typically evaluated with a set of scenarios predefined by the authors. For instance, (Lin and Goodrich, 2009) test their algorithm over three single UAV scenarios (whose initial beliefs are composed by a single Gaussian, two separated Gaussians and two

overlapping Gaussians), (Berger et al., 2016) analyze their approach over two static scenarios (with exponential and uniform beliefs) using 5 UAVs and lastly, this thesis analyzes the algorithms performance over several scenarios with different number of UAVs and whose beliefs are composed by several static or dynamic Gaussians. The majority of PS works do not specify a methodology for building their initial beliefs. One interesting exception is the work presented in (San Juan et al., 2018), which as well as proposing several PS algorithms it describes a fuzzy logic approach to build initial probability maps for SaR missions. However, there are several works in the literature that focus on modeling the target location and motion beliefs (Breivik and Allen, 2008; Lin and Goodrich, 2010) or that propose software tools (BMT Group Ltd; Wysokiński et al., 2014) for specific types of search missions. For instance, (Lin and Goodrich, 2010) models the person behavior in Wilderness SaR missions, (Breivik and Allen, 2008) proposes probabilistic target models for maritime SaR missions, (BMT Group Ltd) describes a software tool used for maritime SaR missions and finally, (Wysokiński et al., 2014) describes a software destined for assisting the Canadian Forces during the planning of search missions for finding lost aircrafts. The methods for building the target models described in all of them are restricted to specific mission types and only (Wysokiński et al., 2014) proposes search plans. Inspired in these works, we propose the following simple methodology that allows to construct the initial belief for a variety of scenarios.

To construct the initial target belief $b(v^0)$, we merge knowledge coming from different sources, using a different probability layer $b^l(v^0)$ for each information source and performing with Equation 6.1 an addition of the L probability layers weighted with their reliability/importance coefficients w_l. In other words, considering the first term in Equation 6.1 a normalization coefficient that ensures that $b(v^0)$ is a probability function (i.e. $\sum_{v^0 \in G_\Omega} b(v^0) = 1$), our initial target belief $b(v^0)$ is calculated as the mixture of the beliefs $b^l(v^0)$ associated to the different l-th information sources.

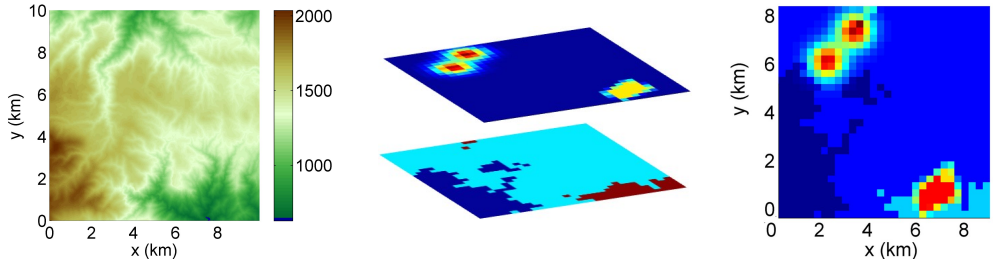
$$b(v^0) = \frac{1}{\sum_{l=1}^{L} w_l} \sum_{l=1}^{L} w_l b^l(v^0) \tag{6.1}$$

For instance, a probability layer $b^l(v^0)$ can be associated to geographical information (e.g. road maps, the terrain altitude or topography) or to intelligence/user-defined clues (e.g. the last or habitual location areas of the target). For building the initial beliefs used during the integration process we consider the following two probability layers:

The terrain elevation probability layer $b^1(v^0)$ contains information about the target location related to the elevation of the terrain within the search area Ω. To define $b^1(v^0)$ first the Digital Elevation Model (DEM, Li et al. (2004)) of Ω is automatically resampled to the size of the cells of G_Ω in order to obtain the average altitude of each cell of G_Ω. Next, the user/operator is required to divide the existing elevations within the cells in G_Ω in consecutive ranges and to assign a chance of target presence to each range. Finally, the method automatically determines the cells in each elevation range and the probability associated to all $v^0 \in G_\Omega$, distributing the chances of target presence assigned by the operator. In this way, the procedure automatically spreads uniformly the belief over different regions of the search area generating a geographical probability layer $b^1(v^0)$, where areas with similar altitude share the same initial belief.

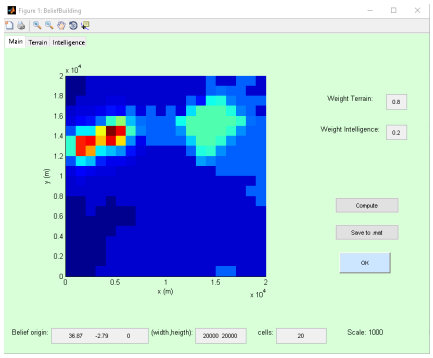
The intelligence probability layer $b^2(v^0)$ contains information about the target location provided by the operator/user. In order to construct this layer, the operator has to define graphically a mixture (weighted addition) of Gaussians (centered in eligible locations of the search area Ω and spread according to eligible standard deviations) and of polygonal areas (defined by their external points placed in the desired locations of Ω) with uniform probabilities (assigned by the operator). The weights of each element (each Gaussian and/or each polygonal area) within the intelligence probability layer $b^2(v^0)$ are also selected by the operator according to the information gathered about the last know location of the target.

As an example, Figure 6.10 shows the definition procedure of the initial belief of one of the search scenarios considered during the integration process, where an off-road vehicle (target) is lost in Gador mountains (Almería, Spain). Figure 6.10 (a) displays the elevation of the search area, Figure 6.10 (b) the terrain (underneath) and

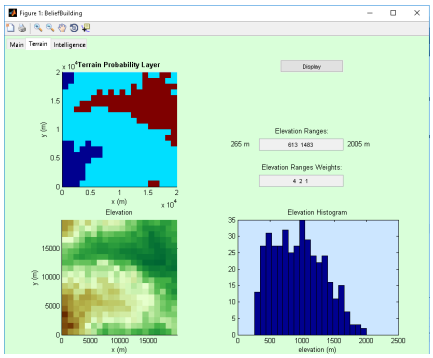


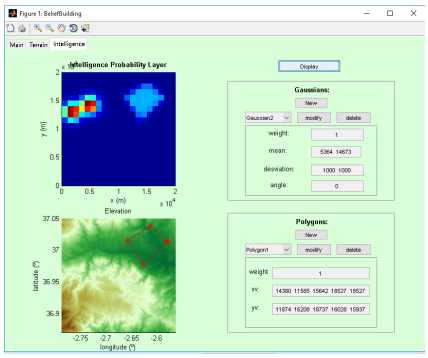
(a) Elevation (m) (b) Probability layers (c) $b(\nu^0)$

Figure 6.10 Scenario for searching for a lost vehicle in Gador mountains (Almería, Spain).



(a) Initial probability map interface





(b) Terrain probability layer interface (c) Intelligence probability layer interface

Figure 6.11 Initial belief construction example using the belief building interface.

intelligence (above) probability layers and Figure 6.10 (c) the resulting probability map, obtained merging both layers with Equation 6.1 considering $w_1 = 0.8$ and $w_2 = 0.2$.

With the purpose of facilitating the labor of the operator, the MTS Planner provides a graphical tool that permits him/her to define the different probability layers

and combine them with the selected reliability coefficients. Figure 6.11 shows the interface of the belief building procedure during the definition of the initial belief of the lost off-road vehicle in Gador mountains. Figure 6.11 (a) displays the main tab, where the initial probability map $b(v^0)$ is shown and which permits modifying the weights of the probability layers, the size of the grid and the dimensions and geographical position of the search area. Figure 6.11 (b) displays the terrain probability layer tab, where the operator can define the elevation ranges and their associated weights. This example considers three different altitude ranges, and as the target is an off-road vehicle, lower altitudes have associated higher chances of target presence. Finally, Figure 6.11 (c) shows the intelligence layer tab, where Gaussian and polygonal identities can be defined. In the example two Gaussian and one polygonal elements are defined in the areas where the target is more likely to be present.

Furthermore, in case that the target is not static the resulting probability map would be modified as time passes according to the target dynamic model. The target dynamic model can also be defined considering intelligence information (e.g. sea currents in case that the target is lost at the sea) and information related to the terrain elevation (e.g. higher probabilities of target presence at the coast). For further information, the reader is referred to (Pérez-Carabaza et al., 2019).

6.5. Results

Once a general view the MTS Planner tool has been described in Section 6.3, this section describes the specifications of the integration processes with two GCS: Airbus R&T and QGroundControl. In addition, this section also presents the integration results obtained during the successive SAVIER project demonstrations.

Both integrations processes (that mainly differ on their communication specifications) follow the same communication flow: the GCS sends information about the search scenario (number of UAVs, entry points to the search area, etc), then the MTS Planner returns an optimized route and, once the mission has started, the GCS sends information about the mission that the MTS Planner monitors and, depending on the

events received, it may re-plan and send a new route. As shown in Figure 6.12, the connection with the two GCS can be established through the *Menu* tab; selecting *MonoUAV* option for connecting with Airbus R&T GCS and *MultiUAV* option for connecting with QGroundControl GCS.

6.5.1. Integration with Airbus R&T GCS

The main components of the Airbus Research & Technology GCS (shown in the photography on the right of Figure 6.17) are the Mission Planning and Monitoring System (MPMS), which permits to define and monitor the mission plan, and the Payload Control and Monitoring System (PCMS), which allows to see the electro-optical sensor output and to control its dynamics. Moreover, Airbus R&T GCS is part of SAVIER Demonstrator, a complex UAS that includes the UAV simulator system of ATLANTE UAV, the software for the generating the synthetic environment (VR-Forces (MÄK Technologies, 2015)) and the software contributions from the different theses funded by SAVIER.

Within this section we chronologically describe the integration process of the MTS Planner with the Airbus R&T GCS and the results that were presented during the SAVIER biannual demonstrations.

6.5.1.1. Phase 0: December 2014

At this early stage of the project we presented a preliminary version of the MTS Planner. This tool has a GUI that allows the user to define MTS scenarios and make use of several MTS algorithms.

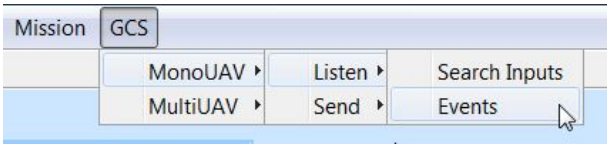
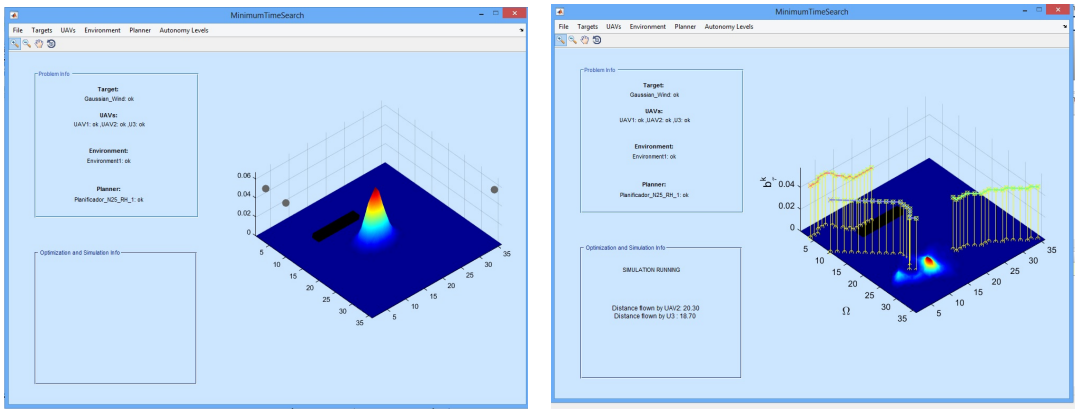


Figure 6.12 GCS tab menu options.



(a) Initial scenario with NFZ in black (b) Proposed search trajectory

Figure 6.13 Preliminary version of the MTS Planner with a search scenario with three UAVs (whose initial positions are displayed with gray circles), a NFZ (represented with a black rectangle) and the search trajectories (displayed with colored lines).

This version of the MTS Planner allows to choose between several MTS algorithms with discrete cardinal motion models (as the ones presented in Chapter 4) and considers ideal sensor models. Figure 6.13 (a) displays the MTS Planner with an initial search scenario with three UAVs and a central gaussian belief and Figure 6.13 (b) shows an example solution for the scenario.

Although this first version allowed to identify the key requirements for a tool that employs the MTS algorithms it has two main drawbacks: it did not provide support to any type of connection with Airbus R&T GCS and the proposed search trajectories (obtained using discrete cardinal motion models for the UAV) were not adequate for a fixed-wing UAV like the ATLANTE.

6.5.1.2. Phase 1: December 2015

In December 2015, we were starting to implement the connections between the MTS Planner and Airbus R&T GCS, and therefore, only a small part of the interaction between both tools was automated.

The configuration of the search scenario within the MTS Planner had to be performed manually, according to the information provided by Airbus of the search

Figure 6.14 Two missions displayed with Google Earth, where the green line represents the route flown by the UAV during the whole mission (while following the orange waypoints defined with Airbus MPMS) and the blue line represents the routes obtained with the MTS Planner, that were discretized to a set of waypoints (represented in blue).

scenarios that will be tested within the GCS. Next, the MTS Planner optimized the search route of the UAV by means of a multi-objective GA based MTS algorithm (proposed in Chapter 5), which considers a continuous UAV dynamic model. The resulting search route was then incorporated to the full mission plan defined by Airbus MPMS. To this end, the search route was discretized to a set of waypoints that were embedded into the full mission plan following the format used by Airbus R&T GCS. In this regard, it was important to compare the UAV motion models within both tools, to make sure that their behavior was similar when the UAV has to follow the same set of waypoints. Figure 6.14 displays some of experiments performed a posteriori with the purpose of comparing the registered route flown by the UAV (displayed in green) when following the mission plan (defined with a set of waypoints) with the route obtained by the MTS Planner (represented in blue).

Results of the Demonstration Figure 6.15 summaries the results of the integration process at this stage. First, the MTS Planner receives the initial mission plan defined by the Airbus MPMS (displayed on the upper left corner of Figure 6.15), which includes the information relative to the search area (represented with a green rectangle). Then, the search scenario is defined manually and optimized by the MTS Planner, which incorporate it to the full mission plan and sends it back to the GCS. Finally, once the GCS has received the full mission plan, the mission is ready to start.

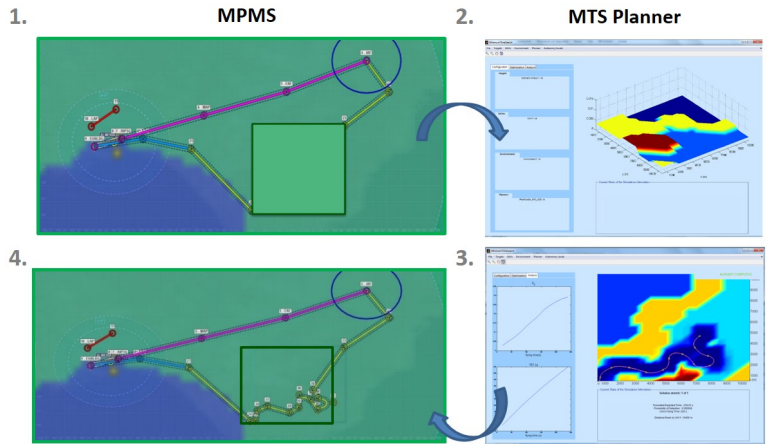


Figure 6.15 Airbus R&T GCS incorporates the search route obtained by the MTS planner.

6.5.1.3. Phase 2: June 2016

At this state of the integration process a bilateral communication between the MTS Planner and Airbus R&T GCS was established following standard communication protocols. On the one hand, the communications are based on the standard STANAG 4586 developed by NATO (Organization, 2012), whose main motivation is to increase the interoperability of the UAS in a way that the GCS can communicate with UAVs from different manufacturers. On the other hand, information sharing and messaging is done through the RTI Data Distribution Service (DDS) (Pardo-Castellote, 2003), which specifies a publish/subscribe middleware and whose objective is the reliable transmission of data in distributed environments with real-time requirements. In DDS the information units (topics or messages) are shared between a publisher of information (writer) and one or more subscribers (readers) through the publish/subscribe paradigm. Figure 6.16 shows some of the topics used for the integration of our MTS Planner with Airbus R&T GCS. Further details about the MTS Planner and the integration process with Airbus R&T GCS are described in (Pérez-Carabaza et al., 2016a).

Results of the Demonstration. The connection between the GCS and the MTS Planner is explained through the steps of an example of connection between the two parts sketched in Figure 6.17. Moreover, each of the connection steps is explained

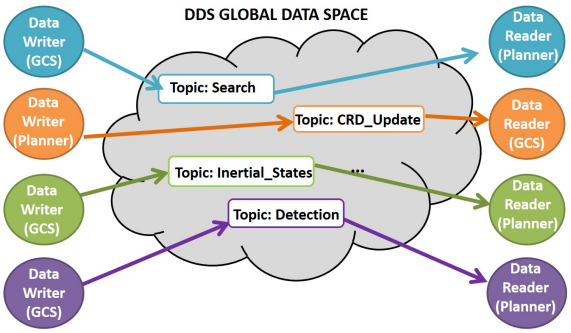


Figure 6.16 Publish/subscribe paradigm in DDS.

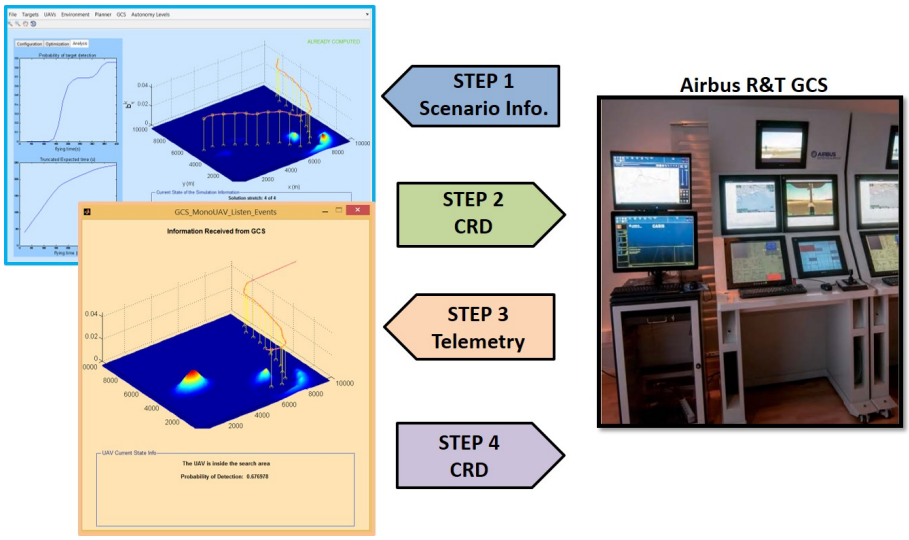


Figure 6.17 Example of the exchange of information between the MTS Planner and the Airbus R&T GCS.

first from a functionality point of view and then from the implementation details. In this way, we first describe a general view of each step which would be common in the process of connection with another GCS, and then explain the implementation details specific of the connection requirements between the MTS Planner and Airbus R&T GCS.

Step 1: The GCS sends the initial information about the search scenario to the MTS Planner. Specifically, the position (latitude, altitude and altitude) of the search area, its dimensions (height, width and rotation angle) and the positions of the UAV ingress and egress points of the search area.

As this information is specific for our MTS Planner, a new message was specified (*Search* topic of Figure 6.16) and sent by the GCS to the MTS Planner via DDS. In order to receive this topic, the operator has to command to start listening to the input information sent by the Airbus R&T GCS through the MTS Planner menu displayed in Figure 6.12 (selecting the following options: *GCS* $\rightarrow$ *MonoUAV* $\rightarrow$ *Listen* $\rightarrow$ *Search Inputs*). Once this option is selected, an executable program, written in C# and designed for letting the MTS Planner listen the *Search* topic, is automatically executed. Once the topic is received its content is displayed in the MTS Planner. After the user accepts the received information, a screen for letting the operator load/create the probability map is displayed. Finally, the optimization of the UAV route is performed with the information contained in the *Search* topic and the probability map selected in the MTS Planner.

Step 2: After the optimization process, the MTS Planner proposes a search route which, if accepted by the operator, is sent to the GCS and incorporated to the mission plan.

Before sending the search route it has to be transformed to the Common Route Definition (CRD) format (xml extension) defined in the STANAG 4586 protocol. To do it, the search route is first discretized into a set of waypoints, each of them with an associated geographical position. Right after sending the search path in the CRD format via TFTP protocol, the topic *CRD_Update* (Figure 6.16) is sent to inform Airbus R&T GCS that there is a new route available that can be integrated to the current mission plan.

Step 3: Once the mission has started, the MTS Planner listens the UAV status information (position) and possible events (sensors detection measurements).

In particular, the MTS Planner receives the UAV state (latitude, longitude, altitude and speed) within the *Inertial_States* message defined in the STANAG 4586 and information about the sensor measurements within the *Detection* message (Figure 6.16). Using this information the planner represents in real-time the UAV position respect the probability map, allowing the operator to monitor the search mission from both the GCS and our MTS Planner and compare the actual route with the planned

search route. An example of the monitoring interface of the MTS Planner is shown in the orange window of Figure 6.17.

Step 4: Depending on the events received, the MTS Planner may re-plan and send new route to the GCS.

Once the target has been detected, a simple re-planning route that consists in flying directly to the egress point (final UAV location) is sent by the MTS Planner. The procedure to send the re-planning route is the same as the one followed for sending the optimized search plan (i.e. the re-planning route is sent via TFTP protocol in CRD format and the DDS topic *CRD_Update* is sent as a confirmation of a new route).

6.5.1.4. Phase 3: January 2017

For January 2017, we incorporated a camera sensor model whose behavior takes into account terrain occlusions and the possibility of considering a predefined sensor pattern during the search mission.

Camera Likelihood with Terrain Occlusion. The sensor model, described by Equation 6.2, considers a likelihood of detecting the target in a given cell $v^t \in G_\Omega$ with the onboard camera equal to the probability of detecting the target within the footprint (P_{camera}) scaled by the percentage of the selected cell within the camera (reducing in this way the probability of detection in the cells of the footprint border).

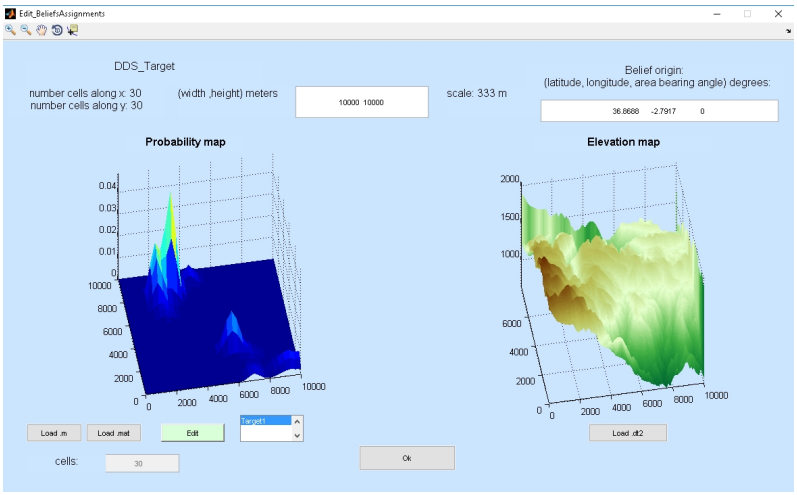
$$P(D_u^t | v^t, s_u^t) = \frac{|v^t \cap footprint(s_u^t)|}{|v^t|} P_{camera} \tag{6.2}$$

where $|v^t|$ is the total area of cell, $|v^t \cap footprint(s_u^t)|$ represents the area (size of the surface) of the common region of the cell and the footprint of the camera (oriented and placed according to s_u^t). The footprint of the sensor is obtained considering the UAV position (x_u^t, y_u^t, h_u^t), the camera azimuth and elevation angles and the terrain altitude (modelled with the National Geospatial-Intelligence Agency Digital Elevation

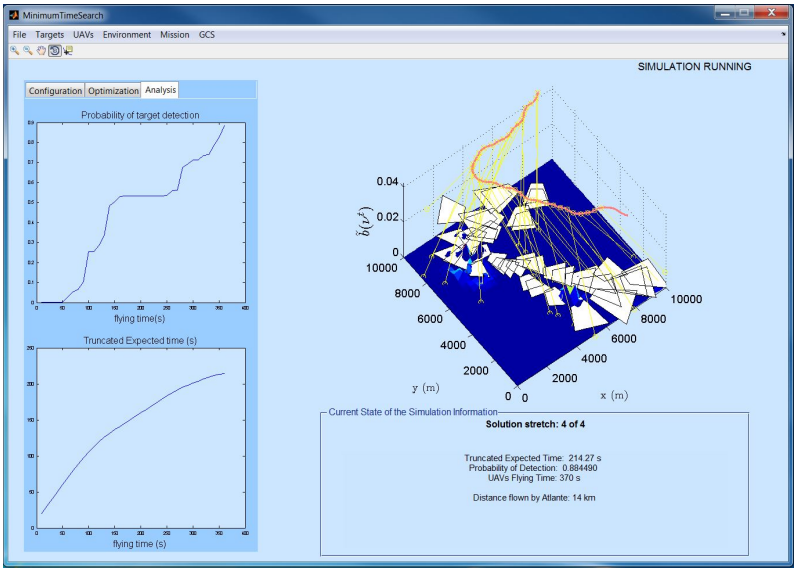
Model (Li et al., 2004)). For further information about the sensor model, the reader is referred to (Pérez-Carabaza et al., 2019).

Predefined sensor search pattern. The possibility of considering a predefined sensor pattern during the search mission permits the UAV not to strictly flight over the higher probability areas to explore them, as they may be reached with the movement of the sensor. Hence, the result of the planner with a sensor pattern may permit to find smoother UAV trajectories than with a fixed down-looking sensor. The search pattern considered for the demo was a S-pattern in azimuth with fixed elevation angle, which was included during the optimization of the UAV trajectory by an extended version of the GA presented in Chapter 5 (as at this stage was more advanced than the ACOR based algorithm). Moreover, with regard to the sensor plan, as the CRD format defined by the STANAG 4586 (which includes the flight route) does not consider the inclusion of a payload plan, the search pattern could not be sent before the mission has started. Our alternative was to make the MTS Planner, once the UAV has entered the search area, start commanding the sensor of Airbus R&T GCS (via DDS through the appropriate messages defined in the STANAG 4586).

Results of the Demonstration. During this demonstration, the search mission consisted in finding a lost vehicle in Gador mountains (Almería, Spain). Figure 6.18 (a) shows the MTS Planner during the definition of the search scenario, whose initial probability map (displayed on the left side) was manually defined considering the terrain map (which is automatically displayed by the planner considering the search area dimensions and geographical position information received from Airbus R&T GCS). Besides, Figure 6.18 (b) displays the solution proposed by the MTS Planner for the search mission, which considers a sensor search S-pattern with the sensor elevation fixed at 45 degrees. To represent the solution we use a red line to show the UAV trajectory, white polygons to display the camera footprints and yellow lines that join the UAV location with the camera aiming point at each measurement without terrain elevation (i.e. at sea level). Hence, in those regions with higher terrain elevations the camera footprint is reduced.



(a) Search scenario in Gador (Almería, Spain)



(b) Proposed search trajectory with predefined sensor S-pattern

Figure 6.18 MTS Planner during (a) the definition of the search scenario and during (b) the simulation of the solution proposed by the planner for searching an off-road vehicle lost in Gador (Almería, Spain).

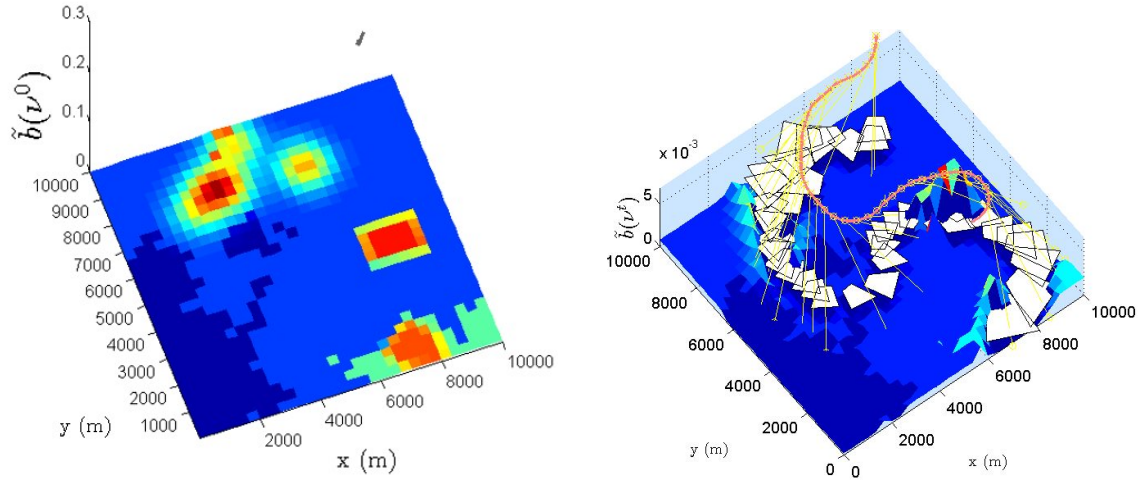
6.5.1.5. Phase 4: September 2017

For December 2017, the two major improvements were 1) the possibility of optimizing the sensor movements and 2) the automatization of probability map building process.

Optimization of the sensor movements. In previous stages of the MTS Planner, the UAV search trajectory was optimized supposing that the sensor stayed with a fixed angle perpendicular to the ground or moved according to a pre-defined sensor pattern. For the final demonstration, we also incorporated the optimization of both the UAV routing and sensor control, enriching in this way the planner with better routes that benefit from the movement of the sensor. This possibility was implemented using an extension of the GA based algorithm proposed in Chapter 5 (as at this stage was more advanced than the ACOR based algorithm), which apart of the UAV heading it also optimizes the sensor azimuth. Further details about the implementation and parametrization of this extended approach can be found in (Pérez-Carabaza et al., 2019).

Automatization of probability map building process. With the purpose of facilitating the labor of the operator, the MTS Planner incorporates the graphical tool presented in Section 6.4, that permits him/her to define the initial target probability map by combining two different probability layers defined from the GUI of the MTS Planner.

Results of the Demonstration. The search mission considered during the last demonstration of the project was again the search for an off-road vehicle lost in Gador mountains, defined using the probability map building tool and process presented in Section 6.4. Furthermore, Figure 6.19 shows the initial probability map (constructed using the probability map building tool) and the search trajectory proposed by the planner. This search trajectory was obtained by means of a GA based MTS algorithm which optimizes the UAV heading and sensor azimuth while keeping constant the sensor elevation of 45 degrees, the UAV height ($h_1^t = 2438$ m) and speed $v_u^t = 76$ m/s. As it can be observed in Figure 6.19 (b), the optimization of the sensor movements allows the UAV to explore the high probability areas without strictly flying over them, reducing in this way the flight time required to observe the high probability areas and obtaining a smoother UAV search trajectory than with a fixed down-looking sensor.



(a) Initial scenario (b) Proposed search trajectory

Figure 6.19 Initial search scenario and proposed solution where the UAV heading and sensor azimuth are optimized with a constant elevation angle of 45 degrees, height $h_1^t = 2438$ m and speed $v_u^t = 76$ m/s. The proposed solution have a length of 15 km, and corresponding $ET(s_1^{1:N}) = 354$ s and $P_d(s_1^{1:N}) = 0.47$.

6.5.2. Integration with QGroundControl GCS

As previously mentioned, the MTS Planner was also integrated with a multi-UAV UAS, which was developed during the project to allow the theses that deal with multiple UAVs test their contributions. This simulated unmanned aerial system uses the open-source QGroundControl (QGC) station to control the UAVs and Unity3D engine to simulate the environment. Further information about its original architecture can be found in (Jesús Ruiz et al., 2016).

In this section we start describing the integration process of the MTS Planner with the QGroundControl GCS and then show some examples of the obtained results, which were presented in the two last demonstrations of SAVIER project (at January and September of 2017).

6.5.2.1. Communication with QGroundControl GCS

The communication flow between the MTS Planner and QGroundControl GCS is similar to the one with Airbus GCS. First, the MTS Planner sets up the search scenario with the information received from QGroundControl. Next, the search trajectories of multiple UAVs are optimized, and after codifying them in a appropriate format they are sent back to the GCS.

An important difference that distinguishes this integration process with QGroundControl GCS from the integration with Airbus R&T GCS is that our planner works in conjunction with the mission planner of other thesis (TD04). Hence, in this case, the higher level mission planner of TD04 thesis optimizes the resources and order of several tasks such as escorting a path, mapping or target search missions and is integrated as a plug-in of the QGroundControl GCS, as detailed in (Ramirez-Atencia and Camacho, 2018). The scenarios of the search mission tasks are sent by the task-mission planner to the MTS Planner, which is in charge of optimizing the UAVs search trajectories and of sending them back to the GCS. Therefore, in contrast to the interaction with Airbus R&T GCS (and the general idea of a planner that was though to optimize a unique search scenario), the MTS Planner can receive from the QGroundControl GCS several search scenarios (assignments), generated by the mission planner of TD04. Moreover, the different assignments received to be optimized may have different or common search areas (e.g. two assignments with the same search area but different UAV entry points).

Communications[3] between the MTS Planner and QGroundControl (QGC) GCS are established through a TCP socket on localhost implemented with the Instrument Control Toolbox of MATLAB. The data transfer between both systems is done with JSON messages, whose structures have been designed ad-hoc for exchanging the required information between the TD04 plug-in of the QGC and the MTS Planner. The communication process and protocol between the MTS Planner and the QGroundControl GCS are summarized in the flowchart of Figure 6.20. First, when

[3]The integration with QGroundControl GCS was developed in collaboration with Julián Bermudez Ortega.

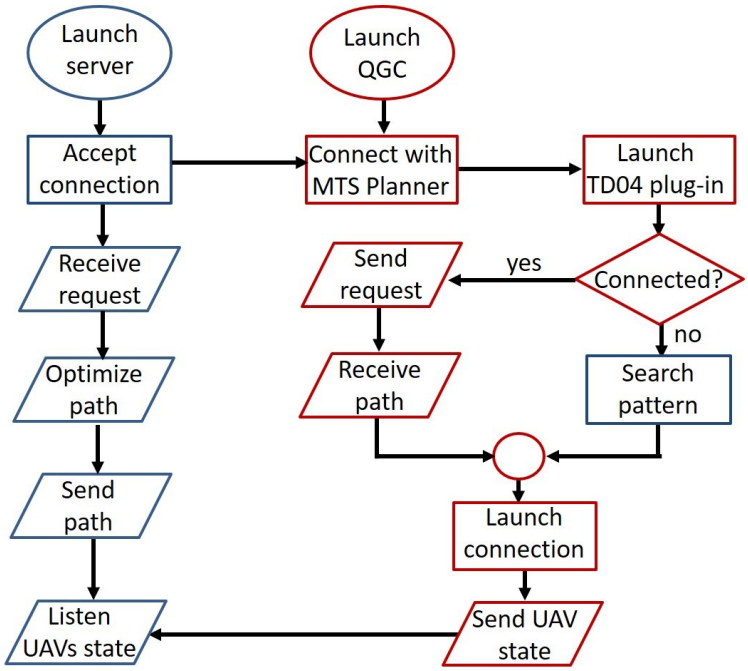


Figure 6.20 Flowchart of connection between the MTS Planner (in blue) and the QGroundControl (GGC) (in maroon).

both application are launched, the MTS planner runs the server and waits until the QGS is connected and QGC tries to connect periodically with the MTS Planner until it succeeds. Next, the TD04 plug-in is launched, and optimizes the whole mission scenario sending to the MTS Planner its requests for optimizing several target search tasks. Once the MTS Planner has finished the optimization of all the assignments, it discretizes the routes to a set of waypoints and format them with an appropriate structure (in JSON format). Moreover, in case that the connection with the MTS Planner is not available, TD04 considers a predefined search pattern. Finally, the MTS Planner sends the solutions to the GCS, the TD04 plug-in receives them and inserts their waypoints into the complete mission plans.

Summarizing, on the one hand, the TD04 Planner generates high level plans for missions that consider several types of tasks taking into account different constraints (such as time restrictions or fuel capacity). On the other hand, the MTS Planner optimizes the routes of the target search task assignments obtained by TD04 planner, which are later incorporated in the high level routes of this planner.

6.5.2.2. Results of the Demonstration

For the integration of our MTS planner with QGC, we present the results of the two demonstrations (at January and September of 2017) together.

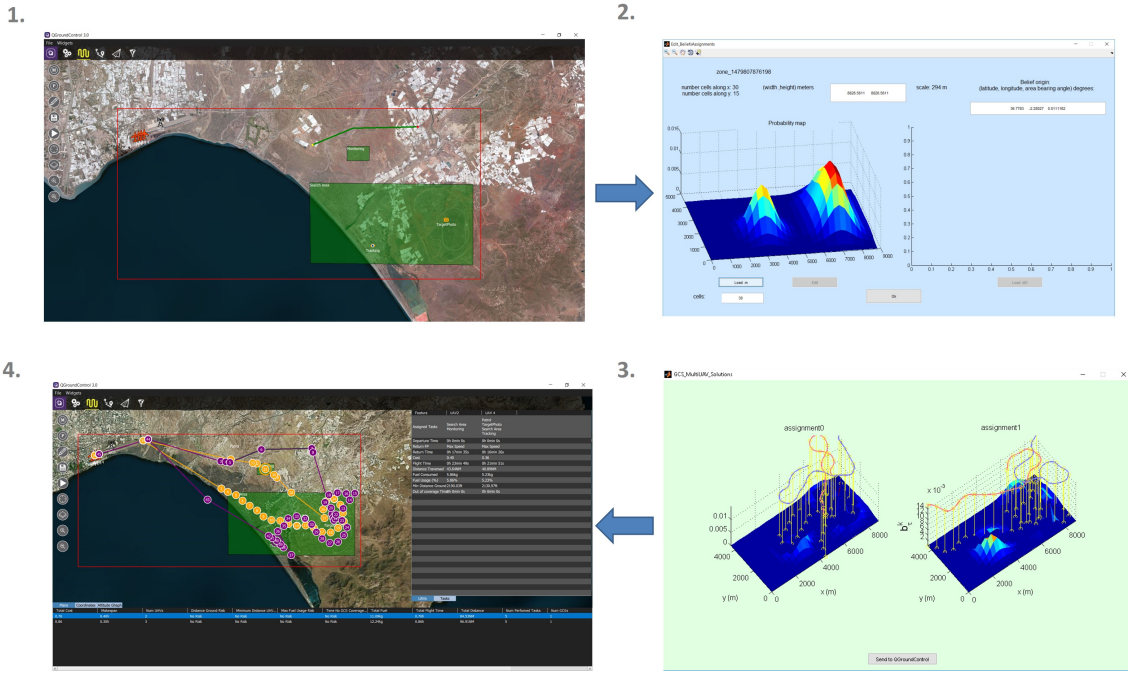


Figure 6.21 Example of initial mission scenario and path planning computed by the mission planner plug-in and MTS Planner. The search area is represented with a green rectangle and the proposed routes with a set of waypoints.

Figure 6.21 shows an example of a plan obtained with the MTS and TD04 Planner working together in QGC. The top left image of Figure 6.21 shows the initial mission scenario used for SAVIER demonstration, which has four UAVs available for performing target searches (green rectangle), escorting (green line) and photography (small green rectangle) tasks. It is worth noting that contrary to the clearly defined search missions used for the demonstrations of the integration with Airbus R&T GCS, in this integration as our planner depends of another stochastic planner, the search scenarios assigned to the MTS Planner may vary. In this example, two search scenarios were assigned to the MTS Planner, both of them consider the same search area and two UAVs, but with different UAV entry points. On the right of the figure is displayed the initial probability map associated to the unique search zone

(and hence the same for the two assignments) and the solutions proposed by the MTS Planner for each of the assignments. Finally, the graphic at the bottom of Figure 6.21 shows one of the solution proposed by TD04 plug-in that uses two of the four UAVs available and which includes the search trajectories sent by our planner. The different metrics of the mission plan considered by TD04 are shown on the right and down informative panels.

6.6. Summary

This chapter summarizes the work done under SAVIER project with the objective of integrating the MTS algorithms capabilities with two different Ground Control Stations (GCS). The project's main motivation was to investigate how different recent technologies can be included and benefit next generation GCS, and encompasses 12 theses that focus on different technologies/problems ranging from dealing with operator stress (Hernando-Gallego and Artés-Rodríguez, 2015) to incorporating augmented reality techniques (Ruano et al., 2017).

One of the main challenges of the project was to include the theses contributions within Airbus R&T GCS, which forms part of the UAS simulator (SAVIER Demostrator) that controls the UAV ATLANTE. Besides, with the purpose of testing the contributions of the theses that deal with multiple UAVs, a multi-UAV freesource UAS that uses QGroundControl GCS was also developed during the project (Jesús Ruiz et al., 2016).

With the purpose of integrating the MTS algorithms proposed in this thesis, we developed a MTS Planner that allows to define the search scenario, to obtain optimized search routes by means of several MTS algorithms and to analyze the proposed solutions. This MTS Planner works as an auxiliary tool that can be used independently or in conjunction with Airbus R&T or QGroundControl GCS. When the planner is used in cooperation with a GCS, it proposes optimized search routes considering the input information received from the GCS and the probability map defined by the operator

(merging topography and intelligence information). The proposed search trajectories are then sent back to the GCS, which incorporates them to the mission plan.

Besides, this chapter also describes the search missions conducted in conjunction with the two GCS and used to show the integration process and the thesis contributions during the biannual demonstrations of SAVIER realized at Airbus headquarters in Getafe.

To sum up, we have identified the requirements that should be considered in order to take advantage of MTS algorithms within a GCS and developed a prototype that facilitates the use of our MTS algorithms by an operator. Finally, the successful integration of the MTS Planner with two different GCS shows the generality of our planner. On the one hand, the MTS Planner is integrated following NATO standards with a complex UAS developed by Airbus. On the other hand, the integration with QGroundControl showed the capabilities of MTS algorithms with multiple UAVs and how the planner can be used in conjunction with a higher level mission planner.

Chapter 7

Conclusions and Future Research Lines

"The important thing is not to stop questioning"

Albert Einstein

This last chapter provides a summary of the main contributions of the work presented in the thesis and discusses some of the possible future research directions.

7.1. Main Conclusions

This thesis tackles the Minimum Time Search (MTS) problem, where a target with uncertain location and dynamics needs to be found as soon as possible. The importance of time, which distinguishes MTS from other target search problems in uncertain environments, is crucial in search missions that look for survivors or that may involve danger (e.g. search for survivors after an earthquake and search for military targets).

The problem is generally tackled from a probabilistic approach and formulated as an optimization problem where the trajectories of the searchers (in our case UAVs) need to be optimized. In the literature we can find numerous algorithmic solutions that propose optimized search routes considering the available uncertain information about the search scenario. The state of the art methods differ on the formulation of

the elements involved in the search (i.e. the target, the searchers and the environment) and on the different characteristics of the optimization methods. For instance, the available information about the possible target locations is generally modeled with a probability map (which indicates the probability of target presence in each of the regions of the discretized search area), but alternatively some works use particle filters. Besides, the search problem has been tackled with a great variety of optimization techniques such as Cross Entropy Optimization, Bayesian Optimization Algorithm or Particle Swarm Optimization. Due to the high computational complexity of the problem, the state of the art approaches assume several simplifications during the modelling of the problem and their optimization techniques are generally approximated methods (that do not ensure finding the optimal solution). Therefore, this thesis aims to propose new efficient MTS algorithms and realistic models of the elements involved in search missions.

In this thesis we tackle the problem from a new approach based on Ant Colony Optimization (ACO) metaheuristic. ACO metaheuristic are iterative algorithms that mimic the intelligence system of ant colonies, where the ants share information through pheromone deposit. At each iteration of ACO a population of ants construct their paths (problem solutions) sampling each solution component from a probability distribution that combines the information of a problem specific heuristic with the information learned from the best solutions of previous iterations (saved in the pheromone table). We have selected the ACO metaheuristic due to its good performance in a variety of high complex problems and its ability to include problem specific information through the use of constructive heuristics. As the results of this thesis have shown, the inclusion of problem specific information is advantageous in a high complex problem like MTS, which requires a good balance between the quality of solutions and the computational time required for obtaining them.

Furthermore, we consider two different formulations of the MTS problem. A discrete version, where the search region is discretized into a grid and the locations of the UAVs are associated to the discretized grid and their control actions take discrete values, and a continuous version, where the locations and control actions of the UAVs

take real values. Therefore, the proposed algorithmic solutions are twofold. Besides, we have also incorporated the MTS algorithms capabilities developed during this thesis with two Ground Control Stations (GCS).

7.1.1. MTS Algorithms for Cardinal UAV Motion Models

We propose two MTS algorithms based on the ACO metaheuristic Max Min Ant System (MMAS), proposed by (Stützle and H. Hoos, 2000), which exploits the knowledge of a new MTS heuristic that guides the UAVs toward the higher and close probability areas. These algorithms propose high-level search trajectories, which consider that the UAVs can move from the centers of the cells of the discretized search region following the cardinal directions. Hence, this type of dynamic is more suitable for rotatory-wing UAVs.

The pheromone encodings of the two proposed algorithms take advantage of the fact that the UAVs can only move from one cell of the grid (node) to the adjacent nodes and hence have more compact pheromone tables that the original ACO encoding. However, they differ in the way the information learned from the best solutions of previous iterations is encoded in the pheromone table. While MMAS-NODE+H learns from the actions that the best found solutions of each iteration have performed at each node (cell), MMAS-TIME+H learns from the actions that the best found solutions of each iteration have performed at each time step. Hence, the former encoding is closer to the original MMAS proposed to solve the Travelling Salesman Problem, where the pheromones learn which movement is the best option at each node (city). By contrast, the latter encoding is more similar to several state of the art methods (e.g. (Lanillos et al., 2012, 2013)), which learn which action is the best option at each time step.

The main conclusions drawn from the analysis of the performance of the proposed MTS MMAS algorithms over several search scenarios and their comparison with several state of the art algorithms are listed below.

- The proposed MTS heuristic allows the algorithms to obtain higher quality solutions from the first iteration and converge faster to higher quality results. To do so, the proposed heuristic gives higher weights (probabilities of been chosen) to the control actions that guide the UAVs toward the closer areas with higher probability of target presence that are still reachable (within the limited time of the mission) from the UAV current position. In order to analyze the positive effects of the MTS heuristic the two proposed algorithms are compared with the performance of their variants without heuristic knowledge (i.e. variants that only use the pheromone information).

- Regarding the two analyzed pheromone encodings, when the heuristic is disabled, learning which actions are the best to perform at each node is a better encoding than learning the best action to take at each time step in some scenarios. However, enabling the MTS heuristic helps MMAS-TIME+H to overcome the drawbacks of its encoding and when the heuristic is enabled the performances of both encodings are similar.

- Although the MTS heuristic by itself is able to construct good quality solutions, the pheromone learning mechanism contributes to improve the performance of the two proposed algorithms in all the analyzed scenarios. In order to analyze the positive effects of the pheromone learning mechanism we have compared the performance of our two proposals with a variant that only considers heuristic information.

- From the comparison analysis of the proposed method against different ad-hoc MTS deterministic heuristic strategies proposed in (Meghjani et al., 2016) and several optimization methods (based on CEO, BOA and GA) previously used for MTS, we can conclude that the ant colony approach presented in this thesis outperforms all the other MTS algorithms, mainly thanks to the use of the proposed MTS heuristic that allows the algorithms to converge faster to higher quality solutions.

7.1.2. MTS Algorithms for Continuous UAV Motion Models

We propose a MTS algorithm based on ant colony optimization for continuous domains (ACOR), proposed by (Socha and Dorigo, 2006), which minimizes the expected target detection time while avoiding overflying restricted areas and collisions among the UAVs. Motivated by the success of ACO algorithms in discrete optimization problems several algorithms that aim to extend ACO to continuous domains have been proposed, among them ACOR claims to maintain the main characteristic of ant based algorithms. The main differences between discrete ACO based algorithms and ACOR are due to the continuous domain of the decision variables (commanded UAV headings in our case) of the latter, that hinder the use of finite tables to save the pheromones. Alternatively, ACOR constructs its solutions (ant tours) sampling each solution component from an archive of best found solutions (equivalent to the pheromone mechanism).

Besides, in this formulation of the problem, we consider a complex radar model and a UAV dynamic model that is adequate to the dynamic restrictions of fixed-wing UAVs. In order to deal with the increased complexity of this formulation we propose two strategies that can be used simultaneously to obtain good solutions quicker. On one side, encouraged by the good results in the discrete MTS formulation, we include a percentage of heuristic ants that use information from a constructive MTS heuristic to build their path (UAVs search trajectories). On the other side, we use a receding horizon controller strategy (dividing the optimization of the full trajectories into several less complex optimization problems) and propose a new optimization criterion which aims to avoid the locality (myopia) problems derived from this approach.

The main conclusions drawn from the analysis of the proposed algorithms over several search scenario are listed below.

- The use of a complex dynamic motion model in the evaluation of the feasibility and optimization criteria permits the algorithm to obtain smooth trajectories that can be followed by fixed-wing UAVs. Besides, the selected radar model has a realistic smooth distance-decaying curve of the probability of detection with the

distance from the UAV to the target, which is common to other type of sensors (e.g. sonars). However, it is worth mentioning that the proposed algorithms in this thesis are not restricted to the selected UAVs models.

- From the conducted experiments we can conclude that the inclusion of heuristic ants increases the solutions quality and the converge speed of the algorithm. In order to analyze the effects of the inclusion of a percentage of heuristic ants we compare the performance of the proposed MTS algorithm (which includes a percentage of heuristic ants) with its variant without heuristic ants and a variant that only considers heuristic ants. As the proposed variant obtained the best results in all the scenarios, we can conclude that both the consideration of heuristic ants and of ACOR's sampling mechanism from the archive of solutions contribute to the good performance of the algorithm.

- From the analyzed experiments we can conclude that the optimization of the proposed myopia reduction criterion allows to avoid the locality of the solutions obtained within a receding horizon controller (multi-step) approach. To do it, the proposed criterion penalizes the trajectories whose final UAV positions are far (and not well oriented) from (towards) the high probability areas of the belief that can be reached by the UAVs in the following optimization steps.

- Finally, from the comparison of the performance of the proposed ACOR based MTS algorithm with a MTS approach based on genetic algorithms, we can conclude that thanks to the inclusion of a percentage of heuristic ants the proposed ACOR based MTS algorithm reaches better results in all the analyzed scenarios.

7.1.3. MTS Planner Integration in Ground Control Station

Furthermore, as part of the initial objectives set by the collaboration project with Airbus that supported this thesis, we have tested the thesis contributions with a Ground Control Station (GCS) developed by Airbus to control its ATLANTE UAV. With this objective in mind, we have developed a graphical user interface that allows

the operator/user to define the scenario, to optimize the search routes by means of several MTS algorithms and to analyze the proposed solutions. Besides, this planner can work independently, in order to analyze the MTS algorithms performance, or as an auxiliary tool in conjunction with two GCS; Airbus GCS and an opensource GCS developed during the project to allow to test contributions of the theses that consider multiple UAVs (Jesús Ruiz et al., 2016).

The successful integration of the MTS Planner with two different GCS shows the generality of our planner. More specifically, the integration with Airbus GCS allowed to test the contributions of this thesis with a complex system developed by Airbus following NATO standards, while the integration with QGroundControl GCS allowed to show the capabilities of MTS algorithms with multiple UAVs and how the planner can be used in conjunction with a higher level mission planner, described in (Ramirez-Atencia and Camacho, 2018). Besides, both integrations follow a similar information flow. First, the operator defines the scenario considering the input information received from the GCS and the probability map (merging topography and intelligence information with the help of a graphical tool). And once the search routes are optimized, the MTS Planner sends the routes to the GCS, which incorporates them to the full mission plan. Finally, during the mission execution, the MTS Planner receives relevant information about the mission in real time in order to monitor and evaluate the mission performance.

7.2. Future Research Lines

Several interesting future research lines of the work presented in this thesis are listed below.

- We consider that the extension of the algorithmic solutions proposed in this thesis to online implementations would be an interesting research direction. In online implementations of MTS algorithms, the computation of the UAV trajectories has to be done during the mission execution and thus the available

computational time is more restricted. Therefore, the MTS heuristics proposed in this thesis would be advantageous for an online implementation of the proposed algorithms due to their accelerating effect in their convergence. Besides, the proposed algorithms have the advantage of being anytime algorithms, which means that they can be terminated at any point and provide a solution (despite not having converged yet). In addition, the computational times of the proposed algorithms could be reduced using more cores of a Graphics Processing Unit (GPU) to achieve real-time computation. Furthermore, although thanks to the increased computational capabilities several online solutions to MTS planning have been proposed in the last years, they only consider the real feedback of the sensors but maintain the basic definition of the search scenario. We think that the possibility of recomputing the search trajectories when there is a change in the number of UAVs (e.g. more UAVs may become available to support the mission) or in the belief (e.g. new information about the target location or dynamics may become available) can be beneficial in certain search missions, and thus, open another possible future line of research.

- Another interesting line of research is the search for multiple heterogeneous targets (e.g. looking for oil leakage and survivors after a disaster or looking for two vehicles moving at different speeds). In general the single-target probabilistic search algorithms can be applied for search of multiple homogeneous targets (which share the same probabilistic models) considering that the search is not finished until all the targets are found. However, in the case of heterogeneous targets is necessary to maintain different probabilistic models for each type of target and the extension of the fitness function is not trivial. For instance, in the state of the art some probabilistic search works consider a different Recursive Bayesian Filter (RBF) for each of the targets and a linear combination of the probability of detection of each target as the fitness criterion (Wong et al., 2005) and (Berger et al., 2014). In the case of MTS, we believe that optimizing the expected time of finding all the targets could be a promising approach.

- Finally, regarding the interaction of the operator with the MTS Planner, we consider as an interesting option the definition of different levels of autonomy, ranging from a low level of autonomy where the operator can modify or propose the search routes, to a high autonomy level where the search mission is automatically performed. The consideration of these different autonomy levels would increase the adaptability of the planner to the autonomy requirements imposed by the system or requested by the operator.

7.3. Research Publications

The journal and conference research publications done during this thesis are listed below:

- Sara Pérez Carabaza, Eva Besada Portas, José Antonio López Orozco and Jesús Manuel de la Cruz, *Resolución del problema de búsqueda en tiempo mínimo mediante colonias de hormigas*, Actas XXXVI Jornadas de Automática, Bilbao 2015.

- Sara Pérez Carabaza, Eva Besada Portas, José Antonio López Orozco and Jesús Manuel de la Cruz, *A real world multi-UAV evolutionary planner for minimum time target detection*, Proceedings of Genetic and Evolutionary Computation Conference (GECCO), Denver 2016. Best paper nomination on the Real World Applications track.

- Sara Pérez Carabaza, Eva Besada Portas, José Antonio López Orozco and Jesús Manuel de la Cruz, *Planificador de búsqueda en tiempo mínimo en un sistema de control de RPAS*, Actas XXXVII Jornadas de Automática, Madrid 2016.

- Sara Pérez Carabaza, Julián Bermúdez Ortega, Eva Besada Portas, José Antonio López Orozco and Jesús Manuel de la Cruz, *A multi-UAV minimum time search planner based on ACOR*, Proceedings of Genetic and Evolutionary Computation Conference (GECCO), Berlín 2017.

- Sara Pérez Carabaza, Eva Besada Portas, José Antonio López Orozco and Jesús Manuel de la Cruz, *Ant colony optimization for multi-UAV minimum time search in uncertain domains*, Journal of Applied Soft Computing, 2018.
- Sara Pérez Carabaza, Eva Besada Portas, José Antonio López Orozco and Gonzalo Pajares, *Minimum time search in real-world scenarios using multiple UAVs with onboard orientable cameras*, Journal of Sensors, 2019.

Bibliography

Benkoski, S. J., Monticino, M. G., and Weisinger, J. R. (1991). A survey of the search theory literature. *Naval Researh Logistic*.

Berger, J., Lo, N., and Barkaoui, M. (2016). Static target search path planning optimization with heterogeneous agents. *Annals of Operations Research*, 244(2):1–18.

Berger, J., Lo, N., and Noel, M. (2014). A new multi-target, multi-agent search-and-rescue path planning approach. *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 8(6):978–987.

Besada-Portas, E., de la Torre, L., Moreno, A., and Risco-Martín, J. L. (2013). On the performance comparison of multi-objective evolutionary UAV path planners. *Information Sciences*, 238:111 – 125.

Bilchev, G. and Parmee, I. C. (1995). *The ant colony metaphor for searching continuous design spaces*, pages 25–39. Springer Berlin Heidelberg.

Blum, C. (2005). Ant colony optimization: Introduction and recent trends. *Physics of Life Reviews*.

BMT Group Ltd. Search and rescue information system (SARIS).

Bourgault, F., Furukawa, T., and Durrant-Whyte, H. F. (2006). *Optimal Search for a Lost Target in a Bayesian World*, pages 209–222. Springer Berlin Heidelberg, Berlin, Heidelberg.

Breivik, O. and Allen, A. (2008). An operational search and rescue model for the norwegian sea and the north sea. *Journal of Marine Research*, 69:99–113.

Budge, M. (2011). Introduction to radar systems. Technical report, University of Alabama in Huntsville.

Carpin, S., Basilico, N., Burch, D., H. Chung, T., and Kölsch, M. (2013). Variable resolution search with quadrotors: Theory and practice. *Journal of Field Robotics*, 30(5):685–701.

Chang-jian, R., Xiao-ming, Q., , and Xu-ning, G. (2015). Distributed cooperative search control method of multiple UAVs for moving target. *International Journal of Aerospace Engineering*.

Chung, T. H., Hollinger, G. A., and Isler, V. (2011). Search and pursuit-evasion in mobile robotics: A survey. *Autonomous Robots*, 31(4):299–316.

Cooper, G. F. and Herskovits, E. (1992). A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347.

Cordón, R. R. (2017). *Future Intensive Use of UAS for Civil and Military Applications in Non-Segregated Airspace-GCS*. PhD thesis, Universidad Politécnica de Madrid.

de la Calle Silos, F. (2017). *Bio-Motivated Features and Deep Learning for Robust Speech Recognition*. PhD thesis, Universidad Carlos III.

Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer methods in applied mechanics and engineering*, 186(2):311–338.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2):182–197.

Delle Fave, F. M., Xu, Z., Rogers, A., and Jennings, N. R. (2010). Decentralised coordination of unmanned aerial vehicles for target search using the max-sum algorithm. In *Proceedings of the Workshop on Agents in Real Time and Environment*, pages 35–44.

Derrac, J., García, S., Molina, D., and Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1):3–18.

Dobbie, J. M. (1974). A two-cell model of search for a moving target. *Operations Research*, 22(1):79–92.

Dorigo, M. and Gambardella, L. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66.

Dorigo, M., Maniezzo, V., and Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 26(1):29–41.

Dréo, J. and Siarry, P. (2002). A new ant colony algorithm using the heterarchical concept aimed at optimization of multiminima continuous functions. In *Proceedings of the International Workshop on Ant Algorithms*, pages 216–221.

Eagle, J. N. (1984). The optimal search for a moving target when the search path is constrained. *Operations Research*, 32(5):1107–1115.

Escario, J. B., Jimenez, J. F., and Giron-Sierra, J. M. (2015). Ant colony extended: Experiments on the travelling salesman problem. *Expert Systems with Applications*, 42(1):390 – 410.

Fan, G. and Jin, S. (2010). Coverage problem in wireless sensor network: A survey. *Journal of Networks*, 5(9):1033–1040.

Feller, W. (1968). *An introduction to probability theory and its applications*. Willey.

Feo, T. A. and Resende, M. G. (1995). Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2):109–133.

Fuhrmann, M. and Horowitz, M. C. (2017). Droning on: explaining the proliferation of unmanned aerial vehicles. *International organization*, 71(2):397–418.

Furukawa, T., Bourgault, F., and Durrant-Whyte, H. F. (2003). Multiple cooperative UAVs engaging multiple targets time-optimally. *IEEE Transactions on Robotics and Automation*.

Furukawa, T., Bourgault, F., Lavis, B., and Durrant-Whyte, H. F. (2006). Recursive bayesian search-and-tracking using coordinated UAVs for lost targets. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2521–2526.

Gaertner, D. and Clark, K. L. (2005). On optimal parameters for ant colony optimization algorithms. In *IC-AI*, pages 83–89.

Gan, S. K. and Sukkarieh, S. (2010). Multi-UAV target search using explicit decentralized gradient-based negotiation. In *Proceedings of the IEEE International Conference on Robotics and Automation*.

Garcia-Najera, A. and Bullinaria, J. A. (2007). Extending ACOR to solve multi-objective problems. In *Proceedings of the UK Workshop on Computational Intelligence*.

Gentile, M. (2015). A theoretical consideration of the parameters of the Max Min Ant System. Master's thesis, Universidad Politecnica de Madrid.

Goss, S., Aron, S., Deneubourg, J.-L., and Pasteels, J. M. (1989). Self-organized shortcuts in the argentine ant. *Naturwissenschaften*, 76(12):579–581.

Goyal, S. (2010). A survey on travelling salesman problem. In *Midwest Instruction and Computing Symposium*, pages 1–9.

Gupta, S. G., Ghonge, M. M., and Jawandhiya, P. (2013). Review of unmanned aircraft system (UAS). *International Journal of Advanced Research in Computer Engineering & Technology*, 2(4):pp–1646.

Hai-bin, D., Xiang-yin, Z., Jiang, W., and Guan-jun, M. (2009). Max-Min adaptive ant colony optimization approach to multi-UAVs coordinated trajectory replanning in dynamic and uncertain environments. *Journal of Bionic Engineering*.

Hernando-Gallego, F. and Artés-Rodríguez, A. (2015). Individual performance calibration using physiological stress signals. *Proceedings of the Workshop of Shimmer sensors, IEEE Body Sensor Networks Conference*.

Hoffmann, G. M., Waslander, S. L., and Tomlin, C. J. (2006). Distributed cooperative search using information-theoretic costs for particle filters, with quadrotor applications. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 21–24.

Holland, J. H. (1992). Genetic algorithms. *Scientific American*, 267:67 – 72.

Interagency Committee on Search and Rescue, I. (1991). National search and rescue manual volume I: National search and rescue system.

Jesús Ruiz, J., Martinez-de Dios, J., Cobano, J. A., and Ollero, A. (2016). A multipayload simulator for cooperative UAS missions. In *International Conference on Unmanned Aircraft Systems*.

Jun, M. and D'Andrea, R. (2003). *Path Planning for Unmanned Aerial Vehicles in Uncertain and Adversarial Environments*, pages 95–110. Springer US.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45.

Khan, A., Yanmaz, E., and Rinner, B. (2015). Information exchange and decision making in micro aerial vehicle networks for cooperative search. *IEEE Transactions on Control of Network Systems*, 2(4):335–347.

Khuller, S., Moss, A., and Naor, J. S. (1999). The budgeted maximum coverage problem. *Information Processing Letters*, 70(1):39–45.

Koopman, B. (1946). Search and screening, operations evaluation group report 56. *Center for Naval Analysis, Alexandria, Virginia*.

Koopman, B. O. (1980). *Search and screening: general principles with historical applications*. Pergamon Press.

Kuhlman, M. J., Otte, M. W., Sofge, D., and Gupta, S. K. (2017). Multipass target search in natural environments. *Sensors*, 17(11):2514.

Lanillos, P. (2013). *Búsqueda de objetivos móviles en tiempo mínimo sobre entornos con incertidumbre*. PhD thesis, Universidad Complutense de Madrid.

Lanillos, P., Besada-Portas, E., Lopez-Orozco, J. A., and de la Cruz, J. M. (2014a). Minimum time search in uncertain dynamic domains with complex sensorial platforms. *Sensors*, 14(8):14131–14179.

Lanillos, P., Besada-Portas, E., Pajares, G., and Ruz, J. J. (2012). Minimum time search for lost targets using cross entropy optimization. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 602–609.

Lanillos, P., Gan, S. K., Besada-Portas, E., Pajares, G., and Sukkarieh, S. (2014b). Multi-UAV target search using decentralized gradient-based negotiation with expected observation. *Information Sciences*, 282:92 –110.

Lanillos, P., Yañez Zuluaga, J., Ruz, J. J., and Besada-Portas, E. (2013). A bayesian approach for constrained multi-agent minimum time search in uncertain dynamic domains. In *Proceeding of the 15th Conference on Genetic and Evolutionary Computation*, pages 391–398.

Lau, H. (2007). *Optimal Search in Structured Environments*. PhD thesis, University of Technology, Sydney.

Li, Z., Zhu, Q., and Gold, C. (2004). *Digital terrain modeling: principles and methodology*.

Lin, L. and Goodrich, M. (2009). UAV intelligent path planning for wilderness search and rescue. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 709–714.

Lin, L. and Goodrich, M. A. (2010). A bayesian approach to modeling lost person behaviors based on terrain features in wilderness search and rescue. *Computational and Mathematical Organization Theory*, 16(3):300–323.

Liu, P., Chen, A. Y., Huang, Y.-N., Han, J.-Y., Lai, J.-S., Kang, S.-C., Wu, T.-H., Wen, M.-C., and Tsai, M.-H. (2014). A review of rotorcraft unmanned aerial vehicle (UAV) developments and applications in civil engineering. *Smart Structures and Systems*, 13(6):1065–1094.

Lo, N., Berger, J., and Noel, M. (2012). Toward optimizing static target search path planning. In *Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defence Application*, pages 1–7.

Madadgar, S. and Afshar, A. (2008). An improved continuous ant algorithm for optimization of water resources problems. *Water Resour Manage*.

Mantecón, T., del Blanco, C. R., Jaureguizar, F., and García, N. (2014). New generation of human machine interfaces for controlling UAV through depth-based gesture recognition. In *Proceeding of the XVI Unmanned Systems Technology Conference*, volume 9084.

Mao, C., Xiao, L., Yu, X., and Chen, J. (2015). Adapting ant colony optimization to generate soft data for software structural testing. *Swarm and Evolutionary Computation*.

Martín, J., Angelina, H., Heredia, G., and Ollero, A. (2016). Tanker UAV for autonomous aerial refueling. In *Proceedings of the Second Iberian Robotics Conference*, pages 571–583. Springer.

McGrayne, S. (2014). The theory that never died: How an eighteenth century mathematical idea transformed the twenty-first century. *Mètode Science Studies Journal-Annual Review*, 5:159–165.

Meghjani, M., Manjanna, S., and Dudek, G. (2016). Multi-target rendezvous search. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 2596–2603.

MÄK Technologies (2015). VR-Forces: Computer generated forces and simulator development.

Nallaperuma, S., Wagner, M., and Neumann, F. (2015). Analyzing the effects of instance features and algorithm parameters for max–min ant system and the traveling salesperson problem. *Frontiers in Robotics and AI*, 2:18.

Neapolitan, R. E. et al. (2004). *Learning bayesian networks*, volume 38. Pearson Prentice Hall Upper Saddle River, NJ.

Newcome, L. R. (2004). *Unmanned aviation: a brief history of unmanned aerial vehicles*. American Institute of Aeronautics and Astronautics.

Organization, N. A. T. (2012). *Interfaces of UAV Control System (UCS) for NATO UAV Interoperability*, 3 edition.

Paniagua Diez, F., Suarez Touceda, D., Sierra Camara, J. M., and Zeadally, S. (2015). Toward self-authenticable wearable devices. *IEEE Wireless Communications*, 22(1):36–43.

Pardo-Castellote, G. (2003). OMG data-distribution service: Architectural overview. In *Proceedings of IEEE 23rd International Conference on Distributed Computing Systems Workshops*, pages 200–206.

Pelikan, M., Goldberg, D. E., and Cantu-Paz, E. (1999). BOA: The bayesian optimization algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 525–532.

Pellegrini, P., Favaretto, D., and Moretti, E. (2006). On max-min ant system's parameters. In *Proceedings of the 5th International Conference on Ant Colony Optimization and Swarm Intelligence*, ANTS'06, pages 203–214, Berlin, Heidelberg. Springer-Verlag.

Perez-Rodriguez, D., Maza, I., Caballero, F., Scarlatti, D., Casado, E., and Ollero, A. (2013). A ground control station for a multi- UAV surveillance system: Design and validation in field experiments. *Journal of Intelligent & Robotic Systems*, 69(1-4):119–130.

Pillai, S. U. and Papoulis, A. (2002). *Probability, random variables, and stochastic processes*, volume 2. McGraw-Hill Times Roman by Science Typographers.

Pollock, S. M. (1970). A simple model of search for a moving target. *Operations Research*, 18(5):883–903.

Pérez-Carabaza, S., Bermúdez-Ortega, J., Besada-Portas, E., López-Orozco, J. A., and de la Cruz, J. M. (2017). A multi-UAV minimum time search planner based on ACOR. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 35–42.

Pérez-Carabaza, S., Besada-Portas, E., López-Orozco, J. A., and de la Cruz, J. M. (2016a). Planificador de búsqueda en tiempo mínimo en un sistema de control de RPAS. In *Proceedings of the XXXVII Jornadas de Automática*.

Pérez-Carabaza, S., Besada-Portas, E., López-Orozco, J. A., and de la Cruz, J. M. (2016b). A real world multi-UAV evolutionary planner for minimum time target detection. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 981–988.

Pérez-Carabaza, S., Besada-Portas, E., López-Orozco, J. A., and Jesús, M. (2018). Ant colony optimization for multi-UAV minimum time search in uncertain domains. *Applied Soft Computing*, 62:789–806.

Pérez-Carabaza, S., Besada-Portas, E., López-Orozco, J. A., and Pajares, G. (2019). Minimum time search in real-world scenarios using multiple UAVs with onboard orientable cameras. *Journal of sensors*.

Pulford, G. (2005). Taxonomy of multiple target tracking methods. *IEE Proceedings-Radar, Sonar and Navigation*, 152(5):291–304.

Raap, M., Meyer-Nieberg, S., Pickl, S., and Zsifkovits, M. (2016). Aerial vehicle search-path optimization a novel method for emergency operations. *Journal of Optimization Theory and Applications*, 172(3):1–19.

Raap, M., Preuß, M., and Meyer-Nieberg, S. (2019). Moving target search optimization – a literature review. *Computers & Operations Research*, 105:132 – 140.

Raja, P. and Pugazhenthi, S. (2012). Optimal path planning of mobile robots: A review. *International Journal of Physical Sciences*, 7(9):1314–1320.

Ramirez-Atencia, C., Bello-Orgaz, G., R-Moreno, M. D., and Camacho, D. (2014). Branching to find feasible solutions in unmanned air vehicle mission planning. In *Intelligent Data Engineering and Automated Learning*, pages 286–294. Springer.

Ramirez-Atencia, C., Bello-Orgaz, G., R-Moreno, M. D., and Camacho, D. (2017). Solving complex multi-UAV mission planning problems using multi-objective genetic algorithms. *Soft Computing*, 21(17):4883–4900.

Ramirez-Atencia, C. and Camacho, D. (2018). Extending QGroundControl for automated mission planning of UAVs. *Sensors*, 18(7):2339.

Rao, B., Gopi, A. G., and Maione, R. (2016). The societal impact of commercial drones. *Technology in Society*, 45:83–90.

Richardson, H. R. and Stone, L. D. (1971). Operations analysis during the underwater search for scorpion. *Naval Research Logistics*, 18(2):141–157.

Robin, C. and Lacroix, S. (2016). Multi-robot target detection and tracking: taxonomy and survey. *Autonomous Robots*, 40(4):729–760.

Rodriguez-Fernandez, V., Menendez, H. D., and Camacho, D. (2015). Design and development of a lightweight multi- UAV simulator. In *Proceedings of the IEEE 2nd International Conference on Cybernetics*, pages 255–260.

Roldán, J. J., del Cerro, J., and Barrientos, A. (2015). A proposal of methodology for multi- UAV mission modeling. In *Proceedings of the 23th Mediterranean Conference on Control and Automation*, pages 1–7.

Roldán, J. J., Olivares-Méndez, M. A., del Cerro, J., and Barrientos, A. (2018). Analyzing and improving multi-robot missions by using process mining. *Autonomous Robots*, 42(6):1187–1205.

Ruano, S., Cuevas, C., Gallego, G., and García, N. (2017). Augmented reality tool for the situational awareness improvement of UAV operators. *Sensors*, 17(2):297.

Rubinstein, R. (1999). The cross-entropy method for combinatorial and continuous optimization. *Methodology And Computing In Applied Probability*, 1(2):127–190.

Ruiz, J. J., Viguria, A., de Dios, J. R. M., and Ollero, A. (2015). Immersive displays for building spatial knowledge in multi-UAV operations. In *Proceedings of the 2015 International Conference on Unmanned Aircraft Systems*, pages 1043–1048.

Russell, S. J. and Norvig, P. (2016). *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,.

San Juan, V., Santos, M., and Andújar, J. M. (2018). Intelligent UAV map generation and discrete path planning for search and rescue operations. *Complexity*.

Sarmiento, A., Murrieta-Cid, R., and Hutchinson, S. (2009). An efficient motion strategy to compute expected-time locally optimal continuous search paths in known environments. *Advanced Robotics*, 23(12-13):1533–1560.

Shi, Y. and Eberhart, R. C. (1999). Empirical study of particle swarm optimization. In *Proceedings of the IEEE Congress on Evolutionary computation*, volume 3, pages 1945–1950.

Skoglar, P. (2007). *UAV path and sensor planning methods for multiple ground target search and tracking-A literature survey*. Linköping University Electronic Press.

Socha, K. and Dorigo, M. (2006). Ant colony optimization for continuous domains. *European Journal of Operational Research*.

Stone, L. (1975). *Theory of optimal search*, volume 118. Elsevier.

Stützle, T. and Dorigo, M. (1999). ACO algorithms for the quadratic assignment problem. In *New Ideas in Optimization*, pages 33–50. McGraw-Hill Ltd.

Stützle, T. and H. Hoos, H. (2000). Max-Min ant system. *Future Generation Computer Systems*, pages 889–914.

Thierens, D. and Bosman, P. A. (2011). Optimal mixing evolutionary algorithms. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 617–624. ACM.

Tisdale, J., Kim, Z., and Hedrick, J. (2009). Autonomous UAV path planning and estimation. *IEEE Robotics Automation Magazine*, 16(2):35–42.

Trummel, K. E. and Weisinger, J. R. (1986). The complexity of the optimal searcher path problem. *Operations Research*, 34(2):324–327.

Wong, E.-M., Bourgault, F., and Furukawa, T. (2005). Multi-vehicle bayesian search for multiple lost targets. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3169–3174.

Wysokiński, M., Marcjan, R., and Dajda, J. (2014). Decision support software for search & rescue operations. *Procedia Computer Science*, 35:776 – 785.

Yang, P., Tang, K., Lozano, J. A., and Cao, X. (2015). Path planning for single unmanned aerial vehicle by separately evolving waypoints. *IEEE Transactions on Robotics*, 31(5):1130–1146.

Yang, Y., Minai, A., and Polycarpou, M. (2002). Decentralized cooperative search in UAV's using opportunistic learning. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*.

Yao, P., Wang, H., and Ji, H. (2017). Gaussian mixture model and receding horizon control for multiple UAV search in complex environment. *Nonlinear Dynamics*, 88(2):903–919.

Zecchin, A. C., Simpson, A. R., Maier, H. R., Leonard, M., Roberts, A. J., and Berrisford, M. J. (2006). Application of two ant colony optimisation algorithms to water distribution system optimisation. *Mathematical and computer modelling*, 44(5-6):451–468.

Zhang, C., Zhen, Z., Wang, D., and Li, M. (2010). UAV path planning method based on ant colony optimization. In *Proceedings of the IEEE Chinese Control and Decision Conference*, pages 3790–3792.